Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Новосибирский государственный технический университет»

Кафедра систем сбора и обработки данных

“УТВЕРЖДАЮ”
ДЕКАН АВТФ
к.т.н. Рева И. Л.
“___”_____ _____ г.

РАБОЧАЯ ПРОГРАММА УЧЕБНОЙ ДИСЦИПЛИНЫ
Теоретическая информатика

Образовательная программа: 12.03.04 Биотехнические системы и технологии, профиль: Биотехнические и робототехнические системы

Курс: 1, семестр : 2

Факультет автоматики и вычислительной техники,

		Семестр
№	**Вид деятельности**	**2**
1	Всего зачетных единиц (кредитов)	3
2	Всего часов	108
3	Всего занятий в контактной форме, час.	61
4	Лекции, час.	18
5	Практические занятия, час.	0
6	Лабораторные занятия, час.	36
7	из них в активной и интерактивной форме, час.	18
8	Аттестация, час.	2
9	Консультации, час.	5
10	Самостоятельная работа, час.	47
11	Виды самостоятельной работы (курсовой проект, курсовая работа, РГЗ, подготовка к контрольной работе)	контр.
12	Вид аттестации	ДЗ

Новосибирск 2017

Рабочая программа составлена на основании федерального государственного образовательного стандарта высшего образования по направлению (специальности): 12.03.04 Биотехнические системы и технологии

ФГОС введен в действие приказом №216 от 12.03.2015 г. , дата утверждения: 08.04.2015 г.

Место дисциплины в структуре учебного плана: Блок 1, вариативная

Рабочая программа разработана на основе компетентностной модели выпускника по направлению (специальности): 12.03.04 Биотехнические системы и технологии

Рабочая программа обсуждена на заседании кафедры

ССОД, протокол заседания кафедры №2/1 от 20.06.2017

Утверждена на совете факультета автоматики и вычислительной техники, протокол № 6 от 21.06.2017

Программу разработал:

старший преподаватель, Квашнина Е. А.

Заведующий кафедрой:

доцент, к.т.н. Прохоренко Е. В.

Ответственный за образовательную программу:

заведующий кафедрой Прохоренко Е. В.

1. Внешние требования

Таблица 1.1

***Компетенция ФГОС*: ОПК.7 способность учитывать современные тенденции развития электроники, измерительной и вычислительной техники, информационных технологий в своей профессиональной деятельности; *в части следующих результатов обучения*:**
з1. знать методы обработки сигналов и изображений, основы анализа случайных данных, методы повышения дешифровочных свойств изображений,
***Компетенция ФГОС*: ОПК.9 способность использовать навыки работы с компьютером, владеть методами информационных технологий, соблюдать основные требования информационной безопасности; *в части следующих результатов обучения*:**
з1. знать технологию работы на ПК в современных операционных средах, основные методы разработки алгоритмов и программ, структуры данных, используемые для представления типовых информационных объектов, типовые алгоритмы обработки данных
з2. обладать базовыми знаниями в области информатики и современных геоинформационных технологий
у1. знать методы построения современных проблемно-ориентированных прикладных программных средств
у4. владеть навыками использования программных средств и работы в компьютерных сетях
***Компетенция ФГОС*: ПК.16 способность разрабатывать инструкции для персонала по эксплуатации технического оборудования и программного обеспечения биомедицинских и экологических лабораторий; *в части следующих результатов обучения*:**
з2. уметь осваивать новые программные средства для профессиональной деятельности
з3. уметь использовать стандартные пакеты прикладных программ для решения практических задач
у1. уметь разрабатывать инструкции для персонала по эксплуатации технического оборудования и программного обеспечения биомедицинских, биометрических и экологических лабораторий

2. Требования НГТУ к результатам освоения дисциплины

Таблица 2.1

Результаты изучения дисциплины по уровням освоения (иметь представление, знать, уметь, владеть)	**Формы организации занятий**
Теоретическая информатика	
ОПК.7.з1 знать методы обработки сигналов и изображений, основы анализа случайных данных, методы повышения дешифровочных свойств изображений,	
1.иметь представление о методах обработки сигналов и изображений, основы генерации и анализа случайных данных	Лекции; Лабораторные работы; Самостоятельная работа
ОПК.9.з1 знать технологию работы на ПК в современных операционных средах, основные методы разработки алгоритмов и программ, структуры данных, используемые для представления типовых информационных объектов, типовые алгоритмы обработки данных	
2. знать технологию работы на ПК в современных операционных средах, основные методы разработки алгоритмов и программ, структуры данных, используемые для представления типовых информационных объектов, типовые алгоритмы обработки данных	Лекции; Самостоятельная работа
ОПК.9.з2 обладать базовыми знаниями в области информатики и современных геоинформационных технологий	
3.знать базовые понятия в области информатики и современных геоинформационных технологий	Лекции; Самостоятельная работа
ОПК.9.у1 знать методы построения современных проблемно-ориентированных прикладных программных средств	
4.иметь опыт построения современных проблемно-ориентированных прикладных программных средств	Лекции; Лабораторные работы; Самостоятельная работа
ОПК.9.у4 владеть навыками использования программных средств и работы в компьютерных сетях	

5.иметь опыт использования программных средств и работы в компьютерных сетях	Лабораторные работы; Самостоятельная работа
ПК.16.з2 уметь осваивать новые программные средства для профессиональной деятельности	
6.знать и уметь осваивать новые программные средства для профессиональной деятельности	Лабораторные работы; Самостоятельная работа
ПК.16.з3 уметь использовать стандартные пакеты прикладных программ для решения практических задач	
7.знать и уметь использовать стандартные пакеты прикладных программ для решения практических задач	Лекции; Лабораторные работы; Самостоятельная работа
ПК.16.у1 уметь разрабатывать инструкции для персонала по эксплуатации технического оборудования и программного обеспечения биомедицинских, биометрических и экологических лабораторий	
8.уметь разрабатывать инструкции , подготавливать документацию по использованию программных средств	Лабораторные работы; Самостоятельная работа

3. Содержание и структура учебной дисциплины

Таблица 3.1

Темы лекций	Активные формы, час.	Часы	Ссылки на результаты обучения	Учебная деятельность
Семестр: 2				
Дидактическая единица: Правила проектирования программ				
1. Общепринятые правила проектирования программ. Типичные ошибки. Хороший стиль программирования.	1	2	2, 3	Студенты прослушивают лекционный материал. В активной форме провдоится обсуждение и моделирование типичных ошибок при составлении программ.
2. Обработка ошибок. Отличия C++ от C.	1	2	2, 3	Посещение лекции
Дидактическая единица: Основы проектирования алгоритмов.				
3. Алгоритмы. Трудоемкость алгоритмов.	0	2	1, 3	Посещение лекции.
4. Сортировки	2	4	1, 3, 4	После прослушивания лекционного материала студентам предлгается в форме обсуждения подсчитать трудоемкости нескольких типовых алгоритмов сортировки и оценить их эффективность для различных объемов данных.
Дидактическая единица: Структурные типы данных				
5. Списки. Виды списков. Циклические и нециклические списки.	0	2	1, 3	Посещение лекции.
6. Работа со списками.	1	2	1, 2, 3, 7	После прослушивания лекции студенты участвуют в обсуждении и написании фрагментов программ по теме лекции.
7. Деревья и графы. Работа с деревьями и графами.	0	2	1, 3	Посещение лекции.

8. Бинарные деревья.	0	2	1, 3	Посещение лекции.

Таблица 3.2

Темы лабораторных работ	Активные формы, час.	Часы	Ссылки на результаты обучения	Учебная деятельность
Семестр: 2				
Дидактическая единица: Основы проектирования алгоритмов.				
1. Базовые алгоритмы. Многомерные массивы.	1	4	1, 4, 5, 6, 7, 8	Получение навыков создания, инициализации и обработки многомерных массивов, передача многомерных массивов в качестве входных параметров функции
2. Сортировки	4	12	1, 4, 5, 6, 7, 8	Получение навыков: - программирования алгоритмов по их описанию; - оценки трудоемкости алгоритмов; -работы с файловой системой; - понимание зависимости трудоемкости алгоритмов от объемов выборки; - понимание особенностей алгоритмов генерации случайных чисел.
Дидактическая единица: Структурные типы данных				
3. Линейные списки	4	12	1, 4, 5, 6, 7, 8	Студенты приобретают практические навыки разработки программ с применением линейных списков.
4. Деревья.	2	4	1, 4, 5, 6, 7, 8	Студенты приобрестают практические навыки разработки программ с применением динамических структур типа "дерево"
5. Графы.	2	4	1, 4, 5, 6, 7, 8	Студент приобретает практические навыки разработки программ с применением динамических структур типа "граф"

Таблица 3.3

Темы для самостоятельного изучения	Активные формы, час.	Часы	Ссылки на результаты обучения	Учебная деятельность
Семестр: 2				
Дидактическая единица: Правила проектирования программ				
1. Правила проектирования программ	0	12	4, 5, 6, 7, 8	Студент самостоятельно изучает материалы по правилам и практикам проектирования компьютерных программ. Осваивает базовые алгоритмы для решения типовых задач при написании программ.

Дидактическая единица: Основы проектирования алгоритмов.				
2. Практикум подсчета трудоемкости алгоритмов.	0	10	1, 2, 4	Студент самостоятельно изычает и практикуетя в аналлизе фрагментов кода для подсчета трудоемкости.
Дидактическая единица: Структурные типы данных				
3. Специфика генерации случайных чисел.	0	10	1, 2, 4	Студенты самостоятельно осваивают материал по особенностям генерации случайных последовательностей.

4. Самостоятельная работа обучающегося

№	Виды самостоятельной работы	Ссылки на результаты обучения	Часы на выполнение	Часы на консультации
Семестр: 2				
1	Контрольные работы	1, 3	5	2
Подготовка к контрольной работе. Решение тестового варианта.: Романов Е. Л. Практикум по программированию на C++ : [учебное пособие] / Е. Л. Романов ; Новосиб. гос. техн. ун-т. - СПб., 2004. - 426, [1] с. : ил.				
2	Подготовка к занятиям	2, 3	10	3
Изучение материалов по теме лабораторной работы.: Романов Е. Л. Практикум по программированию на C++ : [учебное пособие] / Е. Л. Романов ; Новосиб. гос. техн. ун-т. - СПб., 2004. - 426, [1] с. : ил.				
3	Самостоятельное изучение теоретического материала	1, 2, 4, 5, 6, 7, 8	32	0
Студент изучает темы, приведенные в таблице 3.3 : Романов Е. Л. Практикум по программированию на C++ : [учебное пособие] / Е. Л. Романов ; Новосиб. гос. техн. ун-т. - СПб., 2004. - 426, [1] с. : ил.				

5. Технология обучения

Для организации и контроля самостоятельной работы обучающихся, а также проведения консультаций применяются информационно-коммуникационные технологии (табл. 5.1).

Таблица 5.1

Деятельность	Информационно-коммуникационные технологии
Информирование	e-mail
Консультирование	e-mail
Размещение учебных материалов	ЭБС

Таблица 5.2

Активные и интерактивные формы проведения занятий

№	Наименование активных форм
1	Лекция в форме дискуссии
Краткое описание применения: После прочтения теоретической части студентам предлагается совместно с преподавателем проверить действия алгоритмов, подсчета трудоемкости алгоритмов, оценить формирование сложных структурных типов и т.п.	

6. Правила аттестации обучающихся по учебной дисциплине

Для аттестации обучающихся по дисциплине используется балльно-рейтинговая система (БРС), позволяющая выставлять оценки по традиционной шкале и 15-уровневой ECTS. Краткая информация о БРС приведена в табл. 6.1.

Таблица 6.1

Оцениваемые виды деятельности обучающихся	Мин. балл	Максимальный балл
Семестр: 2		
Подготовка к занятиям:	0	
Лекция: Участие в обсуждении	0	5
Лекция: Участие в обсуждении	0	5
Лекция: Участие в обсуждении	0	5
Лекция: Участие в обсуждении	0	5
Лабораторная №1: Выполнение	2	5
Лабораторная №1: Защита	1	5
Лабораторная №2: Выполнение	2	5
Лабораторная №2: Защита	1	5
Лабораторная №3: Выполнение	2	5
Лабораторная №3: Защита	1	5
Лабораторная №4: Выполнение	2	5
Лабораторная №4: Защита	1	5
Лабораторная №5: Выполнение	2	5
Лабораторная №5: Защита	1	5
Контрольные работы:	5	10
Зачет:	10	20

В таблице 6.2 представлено соответствие форм контроля заявляемым требованиям к результатам освоения дисциплины.

Таблица 6.2

Коды компетенций ФГОС	Результаты обучения	Формы контроля		
		Защита Л/Р	Контр. раб.	Зачет
ОПК.7	з1. знать методы обработки сигналов и изображений, основы анализа случайных данных, методы повышения дешифровочных свойств изображений,			+
ОПК.9	з1. знать технологию работы на ПК в современных операционных средах, основные методы разработки алгоритмов и программ, структуры данных, используемые для представления типовых информационных объектов, типовые алгоритмы обработки данных			+
	з2. обладать базовыми знаниями в области информатики и современных геоинформационных технологий		+	+
	у1. знать методы построения современных проблемно-ориентированных прикладных программных средств			+

	у4. владеть навыками использования программных средств и работы в компьютерных сетях	+		+
ПК.16	з2. уметь осваивать новые программные средства для профессиональной деятельности	+		+
	з3. уметь использовать стандартные пакеты прикладных программ для решения практических задач	+		+
	у1. уметь разрабатывать инструкции для персонала по эксплуатации технического оборудования и программного обеспечения биомедицинских, биометрических и экологических лабораторий	+		+

Фонд оценочных средств по дисциплине представлен в приложении № 1 к рабочей программе.

7. Литература

Основная литература

1. Подбельский В. В. Программирование на языке Си : учебное пособие для вузов по направлениям: "Прикладная математика и информатика", "Информатика и вычислительная техника", специальностям "Прикладная математика", "Вычислительные машины, комплексы, системы и сети управления" / В. В. Подбельский, С. С. Фомин. - М., 2007. - 600 с. : ил., табл.

2. Подбельский В. В. Язык Си++ : [учебное пособие для вузов по направлениям "Прикладная математика" и "Вычислительные машины, комплексы, системы и сети"] / В. В. Подбельский. - М., 2007. - 559 с. : ил., табл.

3. Мейерс С. Эффективное использование C++. 50 рекомендаций по улучшению ваших программ и проектов / Скотт Мейерс. - СПб., 2006. - 235 с.. - На обл. авт.: Скотт Майерс.

4. Кнут Д. Э. Искусство программирования. Т. 2 : пер. с англ. / Дональд Э. Кнут ; под общ. ред. Ю. В. Козаченко. - М. [и др.], 2007. - 828 с. : ил.

5. Кнут Д. Э. Искусство программирования. Т. 3 : пер. с англ. / Дональд Э. Кнут ; под общ. ред. Ю. В. Козаченко. - М. [и др.], 2007. - 822 с. : ил.

6. Макконелл Д. Основы современных алгоритмов : учебное пособие по направлению "Информатика и вычислительная техника" / Дж. Макконелл ; пер. с англ. под ред. С. К. Ландо ; доп. М. В. Ульянова. - М., 2006. - 366 с. : ил.

7. Страуструп Б. Язык программирования C++. Специальное издание / Б. Страуструп ; пер. с англ. С. Анисимова и М. Кононова ; под. ред. Ф. Андреева и А. Ушакова. - М., 2005. - 1096 с. : ил.. - С авт. изм. и доп..

8. Кнут Д. Э. Искусство программирования. Т. 1 : учебное пособие / Дональд Э. Кнут. - М., 2011

9. Романов Е. Л. Си/Си ++. От дилетанта до профессионала [Электронный ресурс] : электронное учебное пособие : для 1-2 курсов направления 230100 "Информатика и вычислительная техника / Романов Е. Л. - Новосибирск, 2010. - 1 электрон. опт. диск (CD-ROM). - Режим доступа: http://elibrary.nstu.ru/source?bib_id=vtls000134024. - Загл. с этикетки диска.- Рег. свидетельство №18891.

10. Кнут Д. Э. Искусство программирования. Т. 4, вып. 2 / Дональд Э. Кнут ; [пер. с англ. и ред. Ю. Г. Гордиенко]. - Москва [и др.], 2008. - 145 с. : ил.. - Парал. тит. л. англ..

Дополнительная литература

1. Шилдт Г. Искусство программирования на C++ : [для программистов] / Герберт Шилдт ; [пер. с англ. Т. Коротяевой]. - СПб., 2005. - 474 с. : ил.

2. Макконелл Д. Основы современных алгоритмов : учебное пособие по направлению "Информатика и вычислительная техника": пер. с англ. / Дж. Макконелл. - М., 2004. - 366 с. : ил.

3. Голуб А. И. C & C++. Правила программирования : [пер. с англ.] / Ален. И. Голуб ; под ред. В. Костенко. - М., 1996. - 272 с. : ил.

4. Скиена С. С. Олимпиадные задачи по программированию : руководство по подготовке к соревнованиям ; пер. с англ. / Стивен С. Скиена, Мигель А. Ревилла ; послесл. В. М. Кирюхина. - М., 2005. - 415 с. : ил.

Интернет-ресурсы

1. ЭБС НГТУ : http://elibrary.nstu.ru/

2. ЭБС «Издательство Лань» : https://e.lanbook.com/

3. ЭБС IPRbooks : http://www.iprbookshop.ru/

4. ЭБС "Znanium.com" : http://znanium.com/

5. :

8. Методическое и программное обеспечение

8.1 Методическое обеспечение

1. Романов Е. Л. Практикум по программированию на С++ : [учебное пособие] / Е. Л. Романов ; Новосиб. гос. техн. ун-т. - СПб., 2004. - 426, [1] с. : ил.

8.2 Специализированное программное обеспечение

1 Visual Studio

9. Материально-техническое обеспечение

Компьютерный класс

№	Наименование	Назначение
1	Презентационное оборудование (мультимедиа-проектор, экран, компьютер для управления)	Применяется для презентации теоретического материала, а также демонтрации методов написания программ и алгоритмов
2	Компьютерный класс (Компьютеры объединены в локальную сеть с выходом в Internet)	601 кабинет 7 корпус

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Новосибирский государственный технический университет»

Кафедра систем сбора и обработки данных

“УТВЕРЖДАЮ”
ДЕКАН АВТФ
к.т.н. Рева И. Л.
“___”_____ _____ г.

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

УЧЕБНОЙ ДИСЦИПЛИНЫ
Теоретическая информатика

Образовательная программа: 12.03.04 Биотехнические системы и технологии

Факультет автоматики и вычислительной техники

Новосибирск 2016

Обобщенная структура фонда оценочных средств учебной дисциплины **Теоретическая информатика**

Тема	Код формируемой компетенции	Знания/умения	Контролирующее мероприятие (экзамен, зачет, курсовой проект и т.п.)
Бинарные деревья.	ОПК.7 ОПК.9	з1. знать методы обработки сигналов и изображений, основы анализа случайных данных, методы повышения дешифровочных свойств изображений, з2. обладать базовыми знаниями в области информатики и современных геоинформационных технологий	Зачет Контрольные работы
Деревья и графы. Работа с деревьями и графами.		з1. знать методы обработки сигналов и изображений, основы анализа случайных данных, методы повышения дешифровочных свойств изображений, з2. обладать базовыми знаниями в области информатики и современных геоинформационных технологий	Зачет Контрольные работы
Алгоритмы. Трудоемкость алгоритмов.		з1. знать методы обработки сигналов и изображений, основы анализа случайных данных, методы повышения дешифровочных свойств изображений, з2. обладать базовыми знаниями в области информатики и современных геоинформационных технологий	Зачет Контрольные работы
Списки. Виды списков. Циклические и нециклические списки.		з1. знать методы обработки сигналов и изображений, основы анализа случайных данных, методы повышения дешифровочных свойств изображений, з2. обладать базовыми знаниями в области информатики и современных геоинформационных технологий	Зачет Контрольные работы

Работа со списками.	ОПК.7 ОПК.9 ПК.17/ОУ	з1. знать методы обработки сигналов и изображений, основы анализа случайных данных, методы повышения дешифровочных свойств изображений, з1. знать технологию работы на ПК в современных операционных средах, основные методы разработки алгоритмов и программ, структуры данных, используемые для представления типовых информационных объектов, типовые алгоритмы обработки данных з2. обладать базовыми знаниями в области информатики и современных геоинформационных технологий з3. уметь использовать стандартные пакеты прикладных программ для решения практических задач	Зачет Контрольные работы
Сортировки		з1. знать методы обработки сигналов и изображений, основы анализа случайных данных, методы повышения дешифровочных свойств изображений, з2. обладать базовыми знаниями в области информатики и современных геоинформационных технологий з2. уметь осваивать новые программные средства для профессиональной деятельности з3. уметь использовать стандартные пакеты прикладных программ для решения практических задач у1. знать методы построения современных проблемно-ориентированных прикладных программных средств у1. уметь разрабатывать инструкции для персонала по эксплуатации технического оборудования и программного обеспечения биомедицинских, биометрических и экологических лабораторий у4. владеть навыками использования программных средств и работы в компьютерных сетях	Зачет Контрольные работы Лабораторная
Общепринятые правила проектирования программ. Типичные ошибки. Хороший	ОПК.9	з1. знать технологию работы на ПК в современных операционных средах, основные методы разработки алгоритмов и программ, структуры данных, используемые	Зачет Контрольные работы

Обработка ошибок. Отличия С++ от С.	ОПК.9	з1. знать технологию работы на ПК в современных операционных средах, основные методы разработки алгоритмов и программ, структуры данных, используемые для представления типовых информационных объектов, типовые алгоритмы обработки данных з2. обладать базовыми знаниями в области информатики и современных геоинформационных технологий	Зачет Контрольные работы
Деревья.	ОПК.9 ПК.17/ОУ	з2. уметь осваивать новые программные средства для профессиональной деятельности з3. уметь использовать стандартные пакеты прикладных программ для решения практических задач у1. уметь разрабатывать инструкции для персонала по эксплуатации технического оборудования и программного обеспечения биомедицинских, биометрических и экологических лабораторий у4. владеть навыками использования программных средств и работы в компьютерных сетях	Зачет Лабораторная
Линейные списки		з2. уметь осваивать новые программные средства для профессиональной деятельности з3. уметь использовать стандартные пакеты прикладных программ для решения практических задач у1. уметь разрабатывать инструкции для персонала по эксплуатации технического оборудования и программного обеспечения биомедицинских, биометрических и экологических лабораторий у4. владеть навыками использования программных средств и работы в компьютерных сетях	Зачет Лабораторная
Графы.		з2. уметь осваивать новые программные средства для профессиональной деятельности з3. уметь использовать стандартные пакеты прикладных программ для решения практических задач у4. владеть навыками использования программных средств и работы в компьютерных сетях	Зачет Лабораторная

Базовые алгоритмы. Многомерные массивы.		з2. уметь осваивать новые программные средства для профессиональной деятельности з3. уметь использовать стандартные пакеты прикладных программ для решения практических задач у4. владеть навыками использования программных средств и работы в компьютерных сетях	Лабораторная
Правила проектирования программ	ПК.17/ОУ	з2. уметь осваивать новые программные средства для профессиональной деятельности з3. уметь использовать стандартные пакеты прикладных программ для решения практических задач	Лабораторная

Правила аттестации по дисциплине Теоретическая информатика

Лабораторный практикум

Учебный курс предусматривает лабораторные работы по 5 темам.

После выполнения задания по каждой лабораторной работе, проводится аттестация знаний студента по теме выполненной работы в форме устного опроса. Оценка лабораторной работы складывается из следующих составляющих:

$\mathbf{n}_{\text{общая}}=\mathbf{n}_{\text{выполнение}}+\mathbf{n}_{\text{ответы на вопросы(защита)}}+\mathbf{n}_{\text{дополнительно}}$, где

$\mathbf{n}_{\text{выполнение}}$ - баллы, полученные за полное выполнение лабораторной работы - 5 баллов (обязательное требование для успешной защиты лабораторной работы 2 балла);

$\mathbf{n}_{\text{ответы на вопросы (защита)}}$ - баллы, полученные за ответы на контрольные вопросы по теме работы (от 1 до 5 баллов), задается не менее трех вопросов, каждый из которых оценивается в пределах от 0,5 до 2 баллов;

$\mathbf{n}_{\text{дополнительно}}$ - баллы, полученные за сверх - учебное освоение материала курса (от 0 до 2 баллов).

Для успешной защиты лабораторной работы необходимо набрать от 3 до 10 баллов, из них 2 балла - за полное выполнение работы.

Лекционные занятия. Участие в обсуждении.

После прочтения теоретической части студентам предлагается совместно с преподавателем проверить действия алгоритмов, подсчета трудоемкости алгоритмов, оценить формирование сложных структурных типов и т.д. Работа оценивается в баллах от 0 до 5.

Максимальное количество баллов, которое может студент получить в ходе всех активных лекционных занятий 20.

Комплект заданий для контрольной работы

по дисциплине **Теоретическая информатика**
(наименование дисциплины)

Вариант контрольной работы состоит из 8 заданий, каждое из которых оценивает одну или несколько тем

Задания распределены по темам следующим образом:

Тема	**Номер задания**
Бинарные деревья	**7,8**
Деревья и графы. Работа с деревьями и графами.	**7,8**
Алгоритмы. Трудоемкость алгоритмов.	**6, 8**
Списки. Виды списков. Циклические и нециклические списки.	**7,8**
Работа со списками.	**7,8**
Сортировки	**5, 8**
Общепринятые правила проектирования программ. Типичные ошибки.	**1, 2, 3, 4**
Хороший стиль программирования	**1,4,6,8**
Обработка ошибок. Отличия C++ от C.	**2, 6,8**

Критерии оценки

Задания 1-5, 7 оцениваются 1 баллом.
Задания 6, 8 оцениваются 2 баллами.

- **пороговый** уровень при выполнении контрольной работы составляет 5 баллов
- **базовый** уровень при выполнении контрольной работы составляет 8 баллов
- **продвинутый** уровень при выполнении контрольной работы составляет 10 баллов

Составитель _____ Е.А. Квашнина
(подпись)

«___»_____20 г.

Комплект заданий для зачета

по дисциплине **Теоретическая информатика**
(наименование дисциплины)

Зачет проводится в письменно-устной форме.

На зачете студенту предлагается письменно ответить на два теоретических вопроса. Оценка за ответ на зачете представляет собой среднеарифметическое значение полученных оценок.

При пересдаче зачета по курсу после неудовлетворительной оценки студент может получить оценку не выше Е.

Таблица соответствия между рейтингом в баллах и оценками ECTS и традиционной шкалой (зачтено/незачтено):

Таблица

Диапазон баллов рейтинга	Оценка ECTS	Традиционная шкала оценки
90-100	A+,A,A-	зачтено
80-89	B+,B,B-	зачтено
70-79	C+,C,C-	зачтено
60-69	D+, D, D-	зачтено
53-59	E	зачтено
0-53	FX,F	незачтено

Критерии оценки

- Ответ засчитывается на **пороговом** уровне, если студент дает определение основных понятий, определяет назначение фрагментов алгоритмов, может подсчитать трудоемкость алгоритмов оценка составляет 10 баллов
- Ответ засчитывается на **базовом** уровне, если студент формулирует базовые алгоритмы, содержательно описывает назначение функций, дает характеристику технологических процессов, проводит анализ эффективности алгоритмов, владеет навыками написания программ с использованием сложных типов данных, оценка составляет 15 баллов
- Ответ засчитывается на **продвинутом** уровне, если проводит сравнительный анализ понятий, теорий, подходов, проводит комплексный анализ, выявляет проблемы в представленных фрагментах алгоритмов, может определить оптимальны решения, правильно оценивает трудоемкость алгоритмов, оценка составляет 20 баллов

Зачет считается сданным, если средняя сумма баллов по всем вопросам составляет не менее 10 баллов.
Коэффициент, с которым учитывается полученная сумма баллов в общей оценке по дисциплине составляет 1.

Полный перечень вопросов для зачета по дисциплине:

Теоретическая информатика

1. Обработка ошибок в языке С. Основные способы.
2. Блок try … catch в С++. Роль. Пример применения.
3. Базовые отличия С++ от С.
4. Деревья. Определение. Виды.
5. Деревья. Основные характеристики.
6. Линейные списки. Виды списков Примеры.
7. Односвязные списки.
8. Двусвязные списки.
9. Циклические списки.
10. Нециклические списки.
11. Графы. Основные понятия.
12. Виды графов.
13. Базовые алгоритмы работы с графами.
14. Базовые алгоритмы работы с циклическими списками.
15. Базовые алгоритмы работы с нециклическими списками.
16. Сортировки. Основные виды.
17. Трудоемкость сортировок. Зависимость от объема выборки.
18. Общепринятые правила проектирования программ.
19. Типичные ошибки при написании программ.
20. Основные правила «хорошего стиля» программирования.
21. Бинарные деревья. Основные определения и алгоритмы работы.
22. Деревья и графы. Сравнение алгоритмов работы.
23. Понятие алгоритма. Трудоемкость алгоритмов.
24. Метод подсчета трудоемкости алгоритма.
25. Списки. Виды списков. Циклические и нециклические списки.

Составитель _____ Е.А. Квашнина

(подпись)

«____»_____20 г.

Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«Новосибирский государственный технический университет»
Кафедра систем сбора и обработки данных

Паспорт для зачета

по дисциплине **Теоретическая информатика**
(наименование дисциплины)

1. Методика оценки

Зачет проводится в письменно-устной форме.

На зачете студенту предлагается письменно ответить на два теоретических вопроса. Оценка за ответ на зачете представляет собой среднеарифметическое значение полученных оценок.

При пересдаче зачета по курсу после неудовлетворительной оценки студент может получить оценку не выше Е.

Таблица соответствия между рейтингом в баллах и оценками ECTS и традиционной шкалой (зачтено/незачтено):

Таблица

Диапазон баллов рейтинга	Оценка ECTS	Традиционная шкала оценки
90-100	A+,A,A-	зачтено
80-89	B+,B,B-	зачтено
70-79	C+,C,C-	зачтено
60-69	D+, D, D-	зачтено
53-59	E	зачтено
0-53	FX,F	незачтено

2. Критерии оценки

- Ответ засчитывается на **пороговом** уровне, если студент дает определение основных понятий, определяет назначение фрагментов алгоритмов, может подсчитать трудоемкость алгоритмов оценка составляет 10 баллов
- Ответ засчитывается на **базовом** уровне, если студент формулирует базовые алгоритмы, содержательно описывает назначение функций, дает характеристику технологических процессов, проводит анализ эффективности алгоритмов, владеет навыками написания программ с использованием сложных типов данных, оценка составляет 15 баллов
- Ответ засчитывается на **продвинутом** уровне, если проводит сравнительный анализ понятий, теорий, подходов, проводит комплексный анализ, выявляет проблемы в представленных фрагментах алгоритмов, может определить

оптимальны решения, правильно оценивает трудоемкость алгоритмов, оценка составляет 20 баллов

3. Шкала оценки
Зачет считается сданным, если средняя сумма баллов по всем вопросам составляет не менее 10 баллов.
Коэффициент, с которым учитывается полученная сумма баллов в общей оценке по дисциплине составляет 1.
В общей оценке по дисциплине баллы за зачет учитываются в соответствии с правилами балльно-рейтинговой системы, приведенными в рабочей программе дисциплины.

Полный перечень вопросов для зачета по дисциплине:
Теоретическая информатика

1. Обработка ошибок в языке С. Основные способы.
2. Блок try … catch в С++. Роль. Пример применения.
3. Базовые отличия С++ от С.
4. Деревья. Определение. Виды.
5. Деревья. Основные характеристики.
6. Линейные списки. Виды списков Примеры.
7. Односвязные списки.
8. Двусвязные списки.
9. Циклические списки.
10. Нециклические списки.
11. Графы. Основные понятия.
12. Виды графов.
13. Базовые алгоритмы работы с графами.
14. Базовые алгоритмы работы с циклическими списками.
15. Базовые алгоритмы работы с нециклическими списками.
16. Сортировки. Основные виды.
17. Трудоемкость сортировок. Зависимость от объема выборки.
18. Общепринятые правила проектирования программ.
19. Типичные ошибки при написании программ.
20. Основные правила «хорошего стиля» программирования.
21. Бинарные деревья. Основные определения и алгоритмы работы.
22. Деревья и графы. Сравнение алгоритмов работы.
23. Понятие алгоритма. Трудоемкость алгоритмов.
24. Метод подсчета трудоемкости алгоритма.
25. Списки. Виды списков. Циклические и нециклические списки.

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Новосибирский государственный технический университет»

Кафедра систем сбора и обработки данных

“УТВЕРЖДАЮ”
ДЕКАН АВТФ
к.т.н. Рева И. Л.
“___”_____ _____ г.

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

УЧЕБНОЙ ДИСЦИПЛИНЫ
Теоретическая информатика

Образовательная программа: 12.03.04 Биотехнические системы и технологии

Факультет автоматики и вычислительной техники

Новосибирск 2016

Обобщенная структура фонда оценочных средств учебной дисциплины **Теоретическая информатика**

Тема	Код формируемой компетенции	Знания/умения	Контролирующее мероприятие (экзамен, зачет, курсовой проект и т.п.)
Бинарные деревья.	ОПК.7 ОПК.9	з1. знать методы обработки сигналов и изображений, основы анализа случайных данных, методы повышения дешифровочных свойств изображений, з2. обладать базовыми знаниями в области информатики и современных геоинформационных технологий	Зачет Контрольные работы
Деревья и графы. Работа с деревьями и графами.		з1. знать методы обработки сигналов и изображений, основы анализа случайных данных, методы повышения дешифровочных свойств изображений, з2. обладать базовыми знаниями в области информатики и современных геоинформационных технологий	Зачет Контрольные работы
Алгоритмы. Трудоемкость алгоритмов.		з1. знать методы обработки сигналов и изображений, основы анализа случайных данных, методы повышения дешифровочных свойств изображений, з2. обладать базовыми знаниями в области информатики и современных геоинформационных технологий	Зачет Контрольные работы
Списки. Виды списков. Циклические и нециклические списки.		з1. знать методы обработки сигналов и изображений, основы анализа случайных данных, методы повышения дешифровочных свойств изображений, з2. обладать базовыми знаниями в области информатики и современных геоинформационных технологий	Зачет Контрольные работы

Работа со списками.	ОПК.7 ОПК.9 ПК.17/ОУ	з1. знать методы обработки сигналов и изображений, основы анализа случайных данных, методы повышения дешифровочных свойств изображений, з1. знать технологию работы на ПК в современных операционных средах, основные методы разработки алгоритмов и программ, структуры данных, используемые для представления типовых информационных объектов, типовые алгоритмы обработки данных з2. обладать базовыми знаниями в области информатики и современных геоинформационных технологий з3. уметь использовать стандартные пакеты прикладных программ для решения практических задач	Зачет Контрольные работы
Сортировки		з1. знать методы обработки сигналов и изображений, основы анализа случайных данных, методы повышения дешифровочных свойств изображений, з2. обладать базовыми знаниями в области информатики и современных геоинформационных технологий з2. уметь осваивать новые программные средства для профессиональной деятельности з3. уметь использовать стандартные пакеты прикладных программ для решения практических задач у1. знать методы построения современных проблемно-ориентированных прикладных программных средств у1. уметь разрабатывать инструкции для персонала по эксплуатации технического оборудования и программного обеспечения биомедицинских, биометрических и экологических лабораторий у4. владеть навыками использования программных средств и работы в компьютерных сетях	Зачет Контрольные работы Лабораторная
Общепринятые правила проектирования программ. Типичные ошибки. Хороший	ОПК.9	з1. знать технологию работы на ПК в современных операционных средах, основные методы разработки алгоритмов и программ, структуры данных, используемые	Зачет Контрольные работы

Обработка ошибок. Отличия С++ от С.	ОПК.9	з1. знать технологию работы на ПК в современных операционных средах, основные методы разработки алгоритмов и программ, структуры данных, используемые для представления типовых информационных объектов, типовые алгоритмы обработки данных з2. обладать базовыми знаниями в области информатики и современных геоинформационных технологий	Зачет Контрольные работы
Деревья.	ОПК.9 ПК.17/ОУ	з2. уметь осваивать новые программные средства для профессиональной деятельности з3. уметь использовать стандартные пакеты прикладных программ для решения практических задач у1. уметь разрабатывать инструкции для персонала по эксплуатации технического оборудования и программного обеспечения биомедицинских, биометрических и экологических лабораторий у4. владеть навыками использования программных средств и работы в компьютерных сетях	Зачет Лабораторная
Линейные списки		з2. уметь осваивать новые программные средства для профессиональной деятельности з3. уметь использовать стандартные пакеты прикладных программ для решения практических задач у1. уметь разрабатывать инструкции для персонала по эксплуатации технического оборудования и программного обеспечения биомедицинских, биометрических и экологических лабораторий у4. владеть навыками использования программных средств и работы в компьютерных сетях	Зачет Лабораторная
Графы.		з2. уметь осваивать новые программные средства для профессиональной деятельности з3. уметь использовать стандартные пакеты прикладных программ для решения практических задач у4. владеть навыками использования программных средств и работы в компьютерных сетях	Зачет Лабораторная

Базовые алгоритмы. Многомерные массивы.		з2. уметь осваивать новые программные средства для профессиональной деятельности з3. уметь использовать стандартные пакеты прикладных программ для решения практических задач у4. владеть навыками использования программных средств и работы в компьютерных сетях	Лабораторная
Правила проектирования программ	ПК.17/ОУ	з2. уметь осваивать новые программные средства для профессиональной деятельности з3. уметь использовать стандартные пакеты прикладных программ для решения практических задач	Лабораторная

Правила аттестации по дисциплине Теоретическая информатика

Лабораторный практикум

Учебный курс предусматривает лабораторные работы по 5 темам.

После выполнения задания по каждой лабораторной работе, проводится аттестация знаний студента по теме выполненной работы в форме устного опроса. Оценка лабораторной работы складывается из следующих составляющих:

$\mathbf{n}_{\text{общая}}=\mathbf{n}_{\text{выполнение}}+\mathbf{n}_{\text{ответы на вопросы(защита)}}+\mathbf{n}_{\text{дополнительно}}$, где

$\mathbf{n}_{\text{выполнение}}$ - баллы, полученные за полное выполнение лабораторной работы - 5 баллов (обязательное требование для успешной защиты лабораторной работы 2 балла);

$\mathbf{n}_{\text{ответы на вопросы (защита)}}$ - баллы, полученные за ответы на контрольные вопросы по теме работы (от 1 до 5 баллов), задается не менее трех вопросов, каждый из которых оценивается в пределах от 0,5 до 2 баллов;

$\mathbf{n}_{\text{дополнительно}}$ - баллы, полученные за сверх - учебное освоение материала курса (от 0 до 2 баллов).

Для успешной защиты лабораторной работы необходимо набрать от 3 до 10 баллов, из них 2 балла - за полное выполнение работы.

Лекционные занятия. Участие в обсуждении.

После прочтения теоретической части студентам предлагается совместно с преподавателем проверить действия алгоритмов, подсчета трудоемкости алгоритмов, оценить формирование сложных структурных типов и т.д. Работа оценивается в баллах от 0 до 5.

Максимальное количество баллов, которое может студент получить в ходе всех активных лекционных занятий 20.

Комплект заданий для контрольной работы

по дисциплине **Теоретическая информатика**
(наименование дисциплины)

Вариант контрольной работы состоит из 8 заданий, каждое из которых оценивает одну или несколько тем

Задания распределены по темам следующим образом:

Тема	**Номер задания**
Бинарные деревья	**7,8**
Деревья и графы. Работа с деревьями и графами.	**7,8**
Алгоритмы. Трудоемкость алгоритмов.	**6, 8**
Списки. Виды списков. Циклические и нециклические списки.	**7,8**
Работа со списками.	**7,8**
Сортировки	**5, 8**
Общепринятые правила проектирования программ. Типичные ошибки.	**1, 2, 3, 4**
Хороший стиль программирования	**1,4,6,8**
Обработка ошибок. Отличия C++ от C.	**2, 6,8**

Критерии оценки

Задания 1-5, 7 оцениваются 1 баллом.
Задания 6, 8 оцениваются 2 баллами.

- **пороговый** уровень при выполнении контрольной работы составляет 5 баллов
- **базовый** уровень при выполнении контрольной работы составляет 8 баллов
- **продвинутый** уровень при выполнении контрольной работы составляет 10 баллов

Составитель _____ Е.А. Квашнина
(подпись)

«___»_____20 г.

Комплект заданий для зачета

по дисциплине **Теоретическая информатика**
(наименование дисциплины)

Зачет проводится в письменно-устной форме.

На зачете студенту предлагается письменно ответить на два теоретических вопроса. Оценка за ответ на зачете представляет собой среднеарифметическое значение полученных оценок.

При пересдаче зачета по курсу после неудовлетворительной оценки студент может получить оценку не выше Е.

Таблица соответствия между рейтингом в баллах и оценками ECTS и традиционной шкалой (зачтено/незачтено):

Таблица

Диапазон баллов рейтинга	Оценка ECTS	Традиционная шкала оценки
90-100	A+,A,A-	зачтено
80-89	B+,B,B-	зачтено
70-79	C+,C,C-	зачтено
60-69	D+, D, D-	зачтено
53-59	E	зачтено
0-53	FX,F	незачтено

Критерии оценки

- Ответ засчитывается на **пороговом** уровне, если студент дает определение основных понятий, определяет назначение фрагментов алгоритмов, может подсчитать трудоемкость алгоритмов оценка составляет 10 баллов
- Ответ засчитывается на **базовом** уровне, если студент формулирует базовые алгоритмы, содержательно описывает назначение функций, дает характеристику технологических процессов, проводит анализ эффективности алгоритмов, владеет навыками написания программ с использованием сложных типов данных, оценка составляет 15 баллов
- Ответ засчитывается на **продвинутом** уровне, если проводит сравнительный анализ понятий, теорий, подходов, проводит комплексный анализ, выявляет проблемы в представленных фрагментах алгоритмов, может определить оптимальны решения, правильно оценивает трудоемкость алгоритмов, оценка составляет 20 баллов

Зачет считается сданным, если средняя сумма баллов по всем вопросам составляет не менее 10 баллов.
Коэффициент, с которым учитывается полученная сумма баллов в общей оценке по дисциплине составляет 1.

Полный перечень вопросов для зачета по дисциплине:

Теоретическая информатика

1. Обработка ошибок в языке С. Основные способы.
2. Блок try … catch в С++. Роль. Пример применения.
3. Базовые отличия С++ от С.
4. Деревья. Определение. Виды.
5. Деревья. Основные характеристики.
6. Линейные списки. Виды списков Примеры.
7. Односвязные списки.
8. Двусвязные списки.
9. Циклические списки.
10. Нециклические списки.
11. Графы. Основные понятия.
12. Виды графов.
13. Базовые алгоритмы работы с графами.
14. Базовые алгоритмы работы с циклическими списками.
15. Базовые алгоритмы работы с нециклическими списками.
16. Сортировки. Основные виды.
17. Трудоемкость сортировок. Зависимость от объема выборки.
18. Общепринятые правила проектирования программ.
19. Типичные ошибки при написании программ.
20. Основные правила «хорошего стиля» программирования.
21. Бинарные деревья. Основные определения и алгоритмы работы.
22. Деревья и графы. Сравнение алгоритмов работы.
23. Понятие алгоритма. Трудоемкость алгоритмов.
24. Метод подсчета трудоемкости алгоритма.
25. Списки. Виды списков. Циклические и нециклические списки.

Составитель _____ Е.А. Квашнина

(подпись)

«____»_____20 г.

Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«Новосибирский государственный технический университет»
Кафедра систем сбора и обработки данных

Паспорт для контрольной работы

по дисциплине **Теоретическая информатика**
(наименование дисциплины)

1. Методика оценки

Вариант контрольной работы состоит из 8 заданий, каждое из которых оценивает одну или несколько тем

Задания распределены по темам следующим образом:

Тема	**Номер задания**
Бинарные деревья	**7,8**
Деревья и графы. Работа с деревьями и графами.	**7,8**
Алгоритмы. Трудоемкость алгоритмов.	**6, 8**
Списки. Виды списков. Циклические и нециклические списки.	**7,8**
Работа со списками.	**7,8**
Сортировки	**5, 8**
Общепринятые правила проектирования программ. Типичные ошибки.	**1, 2, 3, 4**
Хороший стиль программирования	**1,4,6,8**
Обработка ошибок. Отличия C++ от C.	**2, 6,8**

2. Критерии оценки

Задания 1-5, 7 оцениваются 1 баллом.
Задания 6, 8 оцениваются 2 баллами.

- **пороговый** уровень при выполнении контрольной работы составляет 5 баллов

- **базовый** уровень при выполнении контрольной работы составляет 8 баллов
- **продвинутый** уровень при выполнении контрольной работы составляет 10 баллов

3. Шкала оценки

Контрольная работа считается сданной, если сумма баллов по всем заданиям составляет не менее 5 баллов (из 10 возможных).

В общей оценке по дисциплине баллы за контрольную работу учитываются в соответствии с правилами балльно-рейтинговой системы, приведенными в рабочей программе дисциплины.

ВАРИАНТ ТЕСТОВЫЙ

1. Вычислить значение переменной a

```
int a;  a = (sizeof (char) || ('t' - 1)) << (2 & 3);
```

2. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b; d = &b[b[1]]; a = *p + *d++;
```

3. Вычислить значение переменной a

```
int a; if (16 - '6' < 1 << 2) a = 1; else a = 0;
```

4. Чему будет равен результат после выполнения цикла:

```
int i,s;
for(i=1,s=1;i<=3;s*=i++);
```

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
int func (int a, int b){if((a+b)<0)return(-(a+b));else
return (a+b);}
void main(){int m=-10, n=5, k; k = func(m/3,n)+6%3;}
```

6. Написать функцию f, возвращающую указатель на минимальный элемент символьной строки.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int val; A *next; };
int  F(A *p){  int n;
for (n=0; p!=NULL; p=p->next, n++);
return(n);}
```

8. Напишите алгоритм (или функцию) для добавления нового элемента в начало односвязного нециклического списка. Элемент списка хранит целое число.

ВАРИАНТ 1

1. Вычислить значение переменной a

```
int a;  a = (sizeof (char) || ('s' - 1)) << (2 & 3);
```

2. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b; d = &b[b[1]]; a = *p + *d++;
```

3. Вычислить значение переменной a

```
int a; if (16 - '7' < 1 << 2) a = 1; else a = 0;
```

4. Можно ли использовать для разветвления такой оператор:

```
if(k=-3){…}; else {…};
```

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
int func(int a, int b){if(a>=b)return(a);else
return(b);}
void main(){int m=10, n=5, k; k = func(m,n)%2;}
```

6. Написать функцию f, возвращающую указатель на минимальный элемент символьной строки.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int val; A *next,*pred; };
A *F(A *p, int n) { for (; n!=0 && p!=NULL; n--,
p=p->next); return p; }
```

8. Напишите функцию (или алгоритм) для добавления нового элемента в конец односвязного нециклического списка. Элемент списка хранит целое число.

ВАРИАНТ 2

1. Вычислить значение переменной a

```
int a;       a = (24 >> 2 << 3) - ('A' + 1) % 'A';
```

2. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = b; d = &b[1] + 1; a = *d - b[1] + *(++p);
```

3. Вычислить значение переменной a

```
int a; if ((2 != 22)*1 == 1) a = 1; else a = 0;
```

4. Для обращения к одному и тому же элементу массива int a[3]={1,2,3}; предложены три варианта, укажите верные: 1)a[1]; 2) *(a+1); 3) *(&a[0]+1)

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
double func(double a){double c; c=1.0/a; return(c);}
void main(){double m=10, n=2, k; k = func(m/2)+2/4;}
```

6. Написать функцию f, заменяющую цифры в символьной строке на точки.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int val; A *next; };
int  F(A *p){  int n;
for (n=0; p!=NULL; p=p->next, n++);return(n);}
```

8. Напишите алгоритм (или функцию) для добавления нового элемента в конец односвязного нециклического списка. Элемент списка хранит вещественное число.

ВАРИАНТ 3

1. Вычислить значение переменной a

```
int a;
a = (15 && 10 & 5) ? sizeof (char) : '9' - '0' << 2;
```

2. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b + b[1]; d = b; a = (*p + 1)/(*d);
```

3. Вычислить значение переменной a

```
int a; if (sizeof (char)%6 == 6) a = 1; else a = 0;
```

4. Отметит ли наличие ошибок компилятор, если программа будет состоять из такого текста:main(){}

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
int func(int a){int c; c=a*a*a;return(c);}
void main(){int m=-10, n=2, k; k = func(m/3)%2+n;}
```

6. Написать функцию f, находящую сумму отрицательных элементов массива вещественных чисел.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int val; A *next; };
A *F(A *p, int n) { for (; n!=0 && p!=NULL; n--,
p=p->next); return p; }
```

8. Напишите алгоритм (или функцию) для добавления нового элемента в начало односвязного нециклического списка. Элемент списка хранит целое число.

ВАРИАНТ 4

1. Вычислить значение переменной a

```
int a;        a = 36 / 12 % 2 > 's' < 1 << 3;
```

2. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = 2 + b; d = 5 + b; a = *d/(*p++);
```

3. Вычислить значение переменной a

```
int a; if (11 | 11 || 11 == 11) a = 1; else a = 0;
```

4. Взаимозаменяемы ли такие конструкции? Какие?

A) if(k=0)…; Б) if(k==0)…; В) if(k)…;

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
int func(int a, int b){int c; c=a%b;return(c);}
void main(){int m=7, n=5, k; k = func(m,n)+3/2;}
```

6. Написать функцию f, сравнивающую длины двух символьных строк.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int val; A *next; };
A *F(A *ph, int v)
{ struct A *q; q->val = v; q->next = ph; ph = q;
return ph; }
```

8. Напишите функцию (или алгоритм) для добавления нового элемента в начало двусвязного нециклического списка. Элемент списка хранит строку символов.

ВАРИАНТ 5

1. Вычислить значение переменной a

```
int a;        a = 5 * 6 & 10 != 5 << 1;
```

2. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b + 2; d = &(*(p + 1)); a = *p + *d++;
```

3. Вычислить значение переменной a

```
int a; if (2 & 2 | 2 && 2) a = 1; else a = 0;
```

4. Подставьте формальный параметр функции:

```
char *s[]={“One”, “Two”,”Three”};
f(s);… void f(…){…}
```

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
double func(double a, double b){return((a+b)/2);}
void main(){double m=10, n=4, k; k = 2*func(m,n);}
```

6. Написать функцию f, вычисляющую факториал заданного целого числа.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int v;  A *l,*r; };
int F( A *p)
{    if (p==NULL) return(0);
return (1 + F(p->r) + F(p->l)); }
```

8. Напишите функцию (или алгоритм) для добавления нового элемента в конец двусвязного нециклического списка. Элемент списка хранит вещественное число.

ВАРИАНТ 6

1. Вычислить значение переменной a

```
int a;        a = 'A' - 'D' + '7' % '5' < (16 >> 2);
```

2. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = b + '2' - '1'; d = p++; a = *p - 'u';
```

3. Вычислить значение переменной a

```
int a; if (0 << !0 < 1 >> !1) a = 1; else a = 0;
```

4. Для обращения к одному и тому же элементу массива int a[3]={1,2,3}; предложены три варианта, укажите верные: 1)a[1]; 2) *(a+1); 3) *(&a[0]+1)

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
double func(double a){double c; c=1.0/a; return(c);}
void main(){double m=10, n=2, k; k = func(m/2)+2/4;}
```

6. Написать функцию f, заменяющую в символьном массиве строчные латинские буквы на прописные.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int k; int v[10]; A *l,*r; };
int F(A *q){ int i,n,m;
if (q==NULL) return 0;
for (n=0,i=0; i<k; i++) n+=q->v[i]);
return n+F(q->l)+F(q->r); }
```

8. Напишите функцию (или алгоритм) для добавления нового элемента в начало односвязного нециклического списка. Элемент списка хранит вещественное число.

ВАРИАНТ 7

1. Вычислить значение переменной a
```
int a;
a = !0 ? 15 : (13 << 1 ? '8' : 2 * sizeof (char));
```
2. Вычислить значение переменной a
```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b; d = &b[*p++]; a = 15/(*p);
```
3. Вычислить значение переменной a
```
int a; if (25/(3 | 2)%2 > 0) a = 1; else a = 0;
```
4. В программе имеется фрагмент:
```
int i,s; for(i=1,s=0;i<=10;s+=i++);
```
перепишите через while

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.
```
int func(int a, int b){if(a<b)return(b);else
return(a);}
void main(){int m=10, n=5, k; k = func(m,n)%2+n/2;}
```
6. Написать функцию f, возвращающую указатель на последний элемент символьной строки.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F
```
struct A { int val; A *next,*pred; };
A *F(A *p, int n) { for (; n!=0 && p!=NULL; n--,
p=p->next); return p; }
```

8. Напишите функцию (или алгоритм) для добавления нового элемента в конец односвязного нециклического списка. Элемент списка хранит целое число.

ВАРИАНТ 8

1. Вычислить значение переменной a
```
int a; a = 'a' > 'b' < 'c' - 52 >> 1 % ('e' - 'a');
```
2. Вычислить значение переменной a
```
char b[] = "Astral Dominae", a, *p, *d;
p = b; d = &b[0]; a = *p >= *d++;
```
3. Вычислить значение переменной a
```
int a; if (15-16 ? 15 == 16:15 !=16) a=1; else a=0;
```
4. Сколько раз выполняются операторы цикла
```
int i=1; while(1) {if(i==3)break; else {i++;
continue;}}
```
5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.
```
int func(int a, int b){if(a>=b)return(a);else
return(b);}
void main(){int m=10, n=5, k; k = func(m,n)%2;}
```
6. Написать функцию f, возвращающую указатель на заданный элемент массива.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F
```
struct A { int val; A *next; };
A *F(A *ph, int v)
{ struct A *q; q = new A; q->val = v; q->next = ph;
ph = q; return ph; }
```
8. Напишите функцию (или алгоритм) для добавления нового элемента в начало двусвязного нециклического списка. Элемент списка хранит строку символов.

ВАРИАНТ 9

1. Вычислить значение переменной a

```
int a;     a = (15 == 0) || (15 != '0' - '5') / 3;
```

2. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b + 4; d = &b[2]; a = *d + *--p;
```

3. Вычислить значение переменной a: **int** a;

```
if ((int)(sizeof (char)) > -1) a = 1; else a = 0;
```

4. Вычислить значение переменной a

```
char s[] = "informatika", a; int i = 3;
a = (s[3] - s['3' - '0']) + s[i++];
```

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
int func(int n){int i,m=1;for(i=0;i<n;i++)m*=2;
return m;}
void main(){int m=10, n=5, k; k = func(m/3)+n%3}
```

6. Написать функцию f для подсчета ненулевых элементов в массиве вещественных чисел.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int v;  A *l,*r; };
int F( A *p)
{    if (p==NULL) return(0);
return (1 + F(p->r) + F(p->l)); }
```

8. Напишите функцию (или алгоритм) для добавления нового элемента в конец двусвязного нециклического списка. Элемент списка хранит вещественное число.

ВАРИАНТ 10

1. Вычислить значение переменной a

```
int a; a = '7' - '5' + (2 << sizeof (char)) % 4 / 2;
```

2. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = &(*(b + 1)); d = ++p; a = *p - *d;
```

3. Вычислить значение переменной a

```
int a; if (13 + 13 >> 13 < 13) a = 1; else a = 0;
```

4. Для обращения к одному и тому же элементу массива int a[3]={1,2,3}; предложены три варианта, укажите верные: 1)a[1]; 2) *(a+1); 3) *(&a[0]+1)

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
int func(int a, int b){if(a<b)return(b);else
return(a);}
void main(){int m=10, n=5, k; k = func(m,n)%2+n/2;}
```

6. Написать функцию f для формирования массива простых чисел, не больших заданного.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int val; A *next,*pred; };
int  F(A *p){  int n;
for (n=0; p!=NULL; p=p->next, n++);
return(n); }
```

8. Напишите функцию (или алгоритм) для добавления нового элемента в начало односвязного нециклического списка. Элемент списка хранит строку символов.

ВАРИАНТ 11

1. Вычислить значение переменной a

```
int a; a = sizeof (char) || ('3' - 1) <= (22 || 3);
```

2. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b; d = &b[b[1]]; a = *p + *d++;
```

3. Вычислить значение переменной a

```
int a; if (55 | 54 | 53 || 1) a = 1; else a = 0;
```

4. Описание массива имеет вид: int a[2][3]={1,2,3,4,5,6};
Если в тексте программы встретится запись a[1], то компилятор воспримет ее как…

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
double func(double a, double b){double c; c=a+b/2; return(a-c);}
void main(){double m=10, n=5, k; k = func(m,n)+3/2.0;}
```

6. Написать функцию f, возвращающую указатель на минимальный элемент символьной строки.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int val; A *next,*pred; };
A *F(A *p, int n) { for (; n!=0 && p!=NULL; n--, p=p->next); return p; }
```

8. Напишите функцию (или алгоритм) для добавления нового элемента в конец односвязного нециклического списка. Элемент списка хранит вещественное число.

ВАРИАНТ 12

1. Вычислить значение переменной a

```
int a;  a = (24 << 2 < 3) - ('A' + 1) % 'B';
```

2. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = b; d = &b[1] + 1; a = *d - b[1] + *(++p);
```

3. Вычислить значение переменной a

```
int a; if (!(0 << 0 | !0) >= 0) a = 1; else a = 0;
```

4. Сколько раз выполняются операторы цикла

```
int i=1; while(1) {if(i==3)break; else {i++; continue;}}
```

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
int func(int a, int b){if(a>=b)return(a);else return(b);}
void main(){int m=10, n=5, k; k = func(m,n)%2;}
```

6. Написать функцию f, заменяющую цифры в символьной строке на точки.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int val; A *next; };
A *F(A *ph, int v)
{ struct A *q; q = new A; q->val = v; q->next = ph; ph = q; return ph; }
```

8. Напишите функцию (или алгоритм) для добавления нового элемента в начало двусвязного нециклического списка. Элемент списка хранит вещественное число.

ВАРИАНТ 13

1. Вычислить значение переменной a

```
int a;
a = (15 & 10 >= 5) ? sizeof (char) : 9 - 0 << 2;
```

2. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b + b[1]; d = b; a = (*p + 1)/(*d);
```

3. Вычислить значение переменной a

```
int a; if ('z' - 'a' > 'a' - 'z') a = 1; else a = 0;
```

4. Подставьте формальный параметр функции:

```
double z; char t='t'; z=f(&t); … double f(…){…}
```

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
int func(int a, int b){int c; c=a%b; return(c);}
void main(){int m=7, n=5, k; k = func(m,n)+3/2;}
```

6. Написать функцию f, находящую сумму отрицательных элементов массива вещественных чисел.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int k; int v[10]; A *l,*r; };
int F(A *q){ int i,n,m;
if (q==NULL) return 0;
for (n=0,i=0; i<k; i++) n+=q->v[i]);
return n+F(q->l)+F(q->r); }
```

8. Напишите функцию (или алгоритм) для добавления нового элемента в конец двусвязного нециклического списка. Элемент списка хранит целое число.

ВАРИАНТ 14

1. Вычислить значение переменной a

```
int a;  a = 36 / 13 % 1 > 's' < sizeof (char) << 3;
```

2. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = 2 + b; d = 5 + b; a = *d/(*p++);
```

3. Вычислить значение переменной a

```
int a; if (1 + 2 + 3 + 4 + 'a' >> 0) a=1; else a= 0;
```

4.Можно ли использовать для разветвления такую конструкцию? Почему? c=(a=0?a:b);

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
int func(int a, int b){int c; c=a+b; return(c/2);}
void main(){int m=10, n=5, k; k = func(m,n)%2;}
```

6. Написать функцию f, сравнивающую длины двух символьных строк.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int val; A *next,*pred; };
A *F(A *p, int n) { for (; n!=0 && p!=NULL; n--,
p=p->next); return p; }
```

8. Напишите функцию (или алгоритм) для добавления нового элемента в конец односвязного нециклического списка. Элемент списка хранит целое число.

ВАРИАНТ 15

1. Вычислить значение переменной a

```
int a;        a = 5 * 6 && 10 == 5 <= 1;
```

2. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b + 2; d = &(*(p + 1)); a = *p + *d++;
```

3. Вычислить значение переменной a

```
int a;
if (1 * 2 * 3 + 1 + 2 + 3 != 1) a = 1; else a = 0;
```

4. Подставьте формальный параметр функции:

```
int a[3]={1,2,3}; int x; x=f(&a[0]);… int f(…){…}
```

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
int func(int a, int b){if(a<b)return(b);else
return(a);}
void main(){int m=10, n=5, k; k = func(m,n)%2+n/2;}
```

6. Написать функцию f, вычисляющую факториал заданного целого числа.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int val; A *next; };
int  F(A *p){  int n;
for (n=0; p!=NULL; p=p->next, n++); return(n);}
```

8. Напишите алгоритм (или функцию) для добавления нового элемента в конец односвязного нециклического списка. Элемент списка хранит вещественное число.

ВАРИАНТ 16

1. Вычислить значение переменной a

```
int a;    a = 's' - 'p' - 17 % 15 < (16 >> 2);
```

2. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = b + '2' - '1'; d = p++; a = *p - 'u';
```

3. Вычислить значение переменной a

```
int a; if (1 >> 2 ? 6 : !(13)) a = 1; else a = 0;
```

4. Подставьте формальный параметр функции:

```
double z; char t='t'; z=f(&t); … double f(…){…}
```

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
int func(int a, int b){int c; c=a/b; return(c);}
void main(){int m=7, n=5, k; k = func(m,n)+3/2;}
```

6. Написать функцию f, заменяющую в символьном массиве строчные латинские буквы на прописные.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int val; A *next; };
A *F(A *p, int n) { for (; n!=0 && p!=NULL; n--,
p=p->next); return p; }
```

8. Напишите алгоритм (или функцию) для добавления нового элемента в начало односвязного нециклического списка. Элемент списка хранит целое число.

ВАРИАНТ 17

1. Вычислить значение переменной a

```
int a;   a = (10 == '10') ? 15 : (13 << 1 ? 8 : 2);
```

2. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b; d = &b[*p++]; a = 15/(*p);
```

3. Вычислить значение переменной a

```
int a; if ('n' / 'g' % 't' * 'u') a = 1; else a = 0;
```

4. Сколько раз выполняются операторы цикла

```
int k=3; while(1) {if(k=1)break; else {k--; }}
```

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
int func(int a, int b){if(a>=b)return(a);else
return(b);}
void main(){int m=10, n=5, k; k = func(m,n)%2;}
```

6. Написать функцию f, возвращающую указатель на последний элемент символьной строки.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int val; A *next; };
A *F(A *ph, int v){ struct A *q; q->val = v; q->next
= ph; ph = q; return ph; }
```

8. Напишите функцию (или алгоритм) для добавления нового элемента в начало двусвязного нециклического списка. Элемент списка хранит строку символов.

ВАРИАНТ 18

1. Вычислить значение переменной a

```
int a;     a = 1 > 2 >= 'c' - 2 << 1 / ('c' - 'a');
```

2. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = b; d = &b[0]; a = *p >= *d++;
```

3. Вычислить значение переменной a

```
int a; if (11 || 12 | (13 && 14)) a = 1; else a = 0;
```

4. Можно ли использовать для разветвления такой оператор:

```
if(k=-3){…}; else {…};
```

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
int func(int a, int b){if(a>=b)return(a);else
return(b);}
void main(){int m=10, n=5, k; k = func(m,n)%2;}
```

6. Написать функцию f, возвращающую указатель на заданный элемент массива.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int v;  A *l,*r; };
int F( A *p) { if (p==NULL) return(0);
return (1 + F(p->r) + F(p->l)); }
```

8. Напишите функцию (или алгоритм) для добавления нового элемента в конец двусвязного нециклического списка. Элемент списка хранит вещественное число.

ВАРИАНТ 19

1. Вычислить значение переменной a

```
int a;     a = ('b' == 'n') && (15 <= 0 - '5') % 3;
```

2. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b + 4; d = &b[2]; a = *d + *--p;
```

3. Вычислить значение переменной a

```
int a; if (1 >> 2 > 3 << 4 < 5) a = 1; else a = 0;
```

4. Для обращения к одному и тому же элементу массива int a[3]={1,2,3}; предложены три варианта, укажите верные: 1)a[1]; 2) *(a+1); 3) *(&a[0]+1)

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
double func(double a){double c; c=1.0/a; return(c);}
void main(){double m=10, n=2, k; k = func(m/2)+2/4;}
```

6. Написать функцию f для подсчета ненулевых элементов в массиве вещественных чисел.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int k; int v[10]; A *l,*r; };
int F(A *q){ int i,n,m;if (q==NULL) return 0;
for (n=0,i=0; i<k; i++) n+=q->v[i]);
return n+F(q->l)+F(q->r); }
```

8. Напишите функцию (или алгоритм) для добавления нового элемента в начало односвязного нециклического списка. Элемент списка хранит вещественное число.

ВАРИАНТ 20

1. Вычислить значение переменной a

```
int a;    a = '5' - '7' + (2 << '2' - '1') / 4 % 2;
```

2. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = &(*(b + 1)); d = ++p; a = *p - *d;
```

3. Вычислить значение переменной a

```
int a; if (!(1 == !(1 || 1))) a = 1; else a = 0;
```

4. Отметит ли наличие ошибок компилятор, если программа будет состоять из такого текста:main(){}

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
int func(int a){int c; c=a*a*a;return(c);}
void main(){int m=-10, n=2, k; k = func(m/3)%2+n;}
```

6. Написать функцию f для формирования массива простых чисел, не больших заданного.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int val; A *next,*pred; };
A *F(A *p, int n) { for (; n!=0 && p!=NULL; n--,
p=p->next); return p; }
```

8. Напишите функцию (или алгоритм) для добавления нового элемента в конец односвязного нециклического списка. Элемент списка хранит целое число.

ВАРИАНТ 21

1. Вычислить значение переменной a

```
int a;    a = (sizeof (char) & (1 != 1)) << (2 & 3);
```

2. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b; d = &b[b[1]]; a = *p + *d++;
```

3. Вычислить значение переменной a

```
int a; if (16 - '6' < 1 << 2) a = 1; else a = 0;
```

4. Взаимозаменяемы ли такие конструкции? Какие?

A) if(k=0)…; Б) if(k==0)…; В) if(k)…;

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
int func(int a, int b){int c; c=a%b;return(c);}
void main(){int m=7, n=5, k; k = func(m,n)+3/2;}
```

6. Написать функцию f, возвращающую указатель на минимальный элемент символьной строки.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int val; A *next; };
A *F(A *ph, int v)
{ struct A *q; q = new A; q->val = v; q->next = ph;
ph = q; return ph; }
```

8. Напишите функцию (или алгоритм) для добавления нового элемента в начало двусвязного нециклического списка. Элемент списка хранит строку символов.

ВАРИАНТ 22

1. Вычислить значение переменной a

```
int a;     a = (24 && 2 & 3) + ('B' - 1) % 'A';
```

2. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = b; d = &b[1] + 1; a = *d - b[1] + *(++p);
```

3. Вычислить значение переменной a

```
int a; if ((2 != 22)*1 == 1) a = 1; else a = 0;
```

4. Подставьте формальный параметр функции:

```
char *s[]={“One”, “Two”,”Three”};
f(s);… void f(…){…}
```

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
double func(double a, double b){return((a+b)/2);}
void main(){double m=10, n=4, k; k = 2*func(m,n);}
```

6. Написать функцию f, заменяющую цифры в символьной строке на точки.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int v;  A *l,*r; };
int F( A *p)
{    if (p==NULL) return(0);
return (1 + F(p->r) + F(p->l)); }
```

8. Напишите функцию (или алгоритм) для добавления нового элемента в конец двусвязного нециклического списка. Элемент списка хранит вещественное число.

ВАРИАНТ 23

1. Вычислить значение переменной a **int** a;

```
a = (55 - 54 - 53) ? sizeof (char)*2 : 2 - (0 << 2);
```

2. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b + b[1]; d = b; a = (*p + 1)/(*d);
```

3. Вычислить значение переменной a

```
int a; if (sizeof (char)%6 == 6) a = 1; else a = 0;
```

4. Для обращения к одному и тому же элементу массива int a[3]={1,2,3}; предложены три варианта, укажите верные: 1)a[1]; 2) *(a+1); 3) *(&a[0]+1)

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
double func(double a){double c; c=1.0/a; return(c);}
void main(){double m=10, n=2, k; k = func(m/2)+2/4;}
```

6. Написать функцию f, находящую сумму отрицательных элементов массива вещественных чисел.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int val; A *next,*pred; };
int  F(A *p){  int n;
for (n=0; p!=NULL; p=p->next, n++);return(n); }
```

8. Напишите функцию (или алгоритм) для добавления нового элемента в начало односвязного нециклического списка. Элемент списка хранит строку символов.

ВАРИАНТ 24

1. Вычислить значение переменной a

```
int a;   a = 2 / 2 % 2 >= '2' < 2 << 2;
```

2. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = 2 + b; d = 5 + b; a = *d/(*p++);
```

3. Вычислить значение переменной a

```
int a; if (11 | 11 || 11 == 11) a = 1; else a = 0;
```

4. В программе имеется фрагмент:

```
int i,s; for(i=1,s=0;i<=10;s+=i++); перепишите через while
```

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
int func(int a, int b){if(a<b)return(b);else return(a);}
void main(){int m=10, n=5, k; k = func(m,n)%2+n/2;}
```

6. Написать функцию f, сравнивающую длины двух символьных строк.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int val; A *next,*pred; };
A *F(A *p, int n) { for (; n!=0 && p!=NULL; n--, p=p->next); return p; }
```

8. Напишите функцию (или алгоритм) для добавления нового элемента в конец односвязного нециклического списка. Элемент списка хранит вещественное число.

ВАРИАНТ 25

1. Вычислить значение переменной a

```
int a; a = 5 - 4 && 10 != (5 | 5) << 1;
```

2. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b + 2; d = &(*(p + 1)); a = *p + *d++;
```

3. Вычислить значение переменной a

```
int a; if (2 & 2 | 2 && 2) a = 1; else a = 0;
```

4. Сколько раз выполняются операторы цикла

```
int i=1; while(1) {if(i==3)break; else {i++; continue;}}
```

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
int func(int a, int b){if(a>=b)return(a);else return(b);}
void main(){int m=10, n=5, k; k = func(m,n)%2;}
```

6. Написать функцию f, вычисляющую факториал заданного целого числа.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int val; A *next; };
A *F(A *ph, int v)
{ struct A *q; q = new A; q->val = v; q->next = ph; ph = q; return ph; }
```

8. Напишите функцию (или алгоритм) для добавления нового элемента в начало двусвязного нециклического списка. Элемент списка хранит вещественное число.

ВАРИАНТ 26

1. Вычислить значение переменной a

```
int a; a = !0 + 6 + '7' / '5' << 1 > 2;
```

2. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = b + '2' - '1'; d = p++; a = *p - 'u';
```

3. Вычислить значение переменной a

```
int a; if (0 << !0 < 1 >> !1) a = 1; else a = 0;
```

4. Вычислить значение переменной a

```
char s[] = "informatika", a; int i = 3;
a = (s[3] - s['3' - '0']) + s[i++];
```

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
int func(int n){int i,m=1;for(i=0;i<n;i++)m*=2;
return m;}
void main(){int m=10, n=5, k; k = func(m/3)+n%3}
```

6. Написать функцию f, заменяющую в символьном массиве строчные латинские буквы на прописные.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int k; int v[10]; A *l,*r; };
int F(A *q)
{ int i,n,m;if (q==NULL) return 0;
for (n=0,i=0; i<k; i++) n+=q->v[i]);
return n+F(q->l)+F(q->r); }
```

8. Напишите функцию (или алгоритм) для добавления нового элемента в конец двусвязного нециклического списка. Элемент списка хранит целое число.

ВАРИАНТ 27

1. Вычислить значение переменной a

```
int a;
a = 10 ? 10 : (10 << sizeof (char) ? 10 : 10 * 10);
```

2. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b; d = &b[*p++]; a = 15/(*p);
```

3. Вычислить значение переменной a

```
int a; if (25/(3 | 2)%2 > 0) a = 1; else a = 0;
```

4. Для обращения к одному и тому же элементу массива int a[3]={1,2,3}; предложены три варианта, укажите верные: 1)a[1]; 2) *(a+1); 3) *(&a[0]+1)

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
int func(int a, int b){if(a<b)return(b);else
return(a);}
void main(){int m=10, n=5, k; k = func(m,n)%2+n/2;}
```

6. Написать функцию f, возвращающую указатель на последний элемент символьной строки.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int val; A *next,*pred; };
A *F(A *p, int n) { for (; n!=0 && p!=NULL; n--, p=p->next);
return p; }
```

8. Напишите функцию (или алгоритм) для добавления нового элемента в конец односвязного нециклического списка. Элемент списка хранит целое число.

ВАРИАНТ 28

1. Вычислить значение переменной a

```
int a; a = 'a' < 'b' < 'c' - 'c' >> 1 % ('b' - 'a');
```

2. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = b; d = &b[0]; a = *p >= *d++;
```

3. Вычислить значение переменной a

```
int a;
if(15 - 16 ? 15 == 16 : 15 != 16) a = 1; else a = 0;
```

4. Описание массива имеет вид: int a[2][3]={1,2,3,4,5,6};
Если в тексте программы встретится запись a[1], то компилятор воспримет ее как…

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
double func(double a, double b){double c; c=a+b/2;
return(a-c);}
void main(){double m=10, n=5, k; k = func(m,n)+3/2.0;}
```

6. Написать функцию f, возвращающую указатель на заданный элемент массива.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int val; A *next; };
int   F(A *p){    int n;
for (n=0; p!=NULL; p=p->next, n++);return(n);}
```

8. Напишите алгоритм (или функцию) для добавления нового элемента в конец односвязного нециклического списка. Элемент списка хранит вещественное число.

ВАРИАНТ 29

1. Вычислить значение переменной a

```
int a; a = (15 >= 0) | (15 != 0 - 5) + 3;
```

2. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b + 4; d = &b[2]; a = *d + *--p;
```

3. Вычислить значение переменной a

```
int a;
if ((int)(sizeof (char)) > -1) a = 1; else a = 0;
```

4. Сколько раз выполняются операторы цикла

```
int i=1; while(1) {if(i==3)break; else {i++; continue;}}
```

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
int func(int a, int b){if(a>=b)return(a);else return(b);}
void main(){int m=10, n=5, k; k = func(m,n)%2;}
```

6. Написать функцию f для подсчета ненулевых элементов в массиве вещественных чисел.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int val; A *next; };
A *F(A *p, int n) { for (; n!=0 && p!=NULL; n--, p=p->next); return p; }
```

8. Напишите алгоритм (или функцию) для добавления нового элемента в начало односвязного нециклического списка. Элемент списка хранит целое число.

ВАРИАНТ 30

1. Вычислить значение переменной a

```
int a; a = 7 - 77 + (7 << sizeof (char)) / 77 % 7;
```

2. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = &(*(b + 1)); d = ++p; a = *p - *d;
```

3. Вычислить значение переменной a

```
int a; if (13 + 13 >> 13 < 13) a = 1; else a = 0;
```

4. Подставьте формальный параметр функции:

```
double z; char t='t'; z=f(&t); … double f(…){…}
```

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
int func(int a, int b){int c; c=a%b; return(c);}
void main(){int m=7, n=5, k; k = func(m,n)+3/2;}
```

6. Написать функцию f для формирования массива простых чисел, не больших заданного.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int val; A *next; };
A *F(A *ph, int v){ struct A *q; q->val = v; q->next = ph; ph = q; return ph; }
```

8. Напишите функцию (или алгоритм) для добавления нового элемента в начало двусвязного нециклического списка. Элемент списка хранит строку символов.

ВАРИАНТ 31

1. Вычислить значение переменной a

```
int a; a = !0 > 0 <= 0 + !0 << 0;
```

2. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b; d = &b[b[1]]; a = *p + *d++;
```

3. Вычислить значение переменной a

```
int a; if (55 | 54 | 53 || 1) a = 1; else a = 0;
```

4.Можно ли использовать для разветвления такую конструкцию? Почему? c=(a=0?a:b);

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
int func(int a, int b){int c; c=a+b; return(c/2);}
void main(){int m=10, n=5, k; k = func(m,n)%2;}
```

6. Написать функцию f, возвращающую указатель на минимальный элемент символьной строки.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int v;  A *l,*r; };
int F( A *p)
{    if (p==NULL) return(0);
return (1 + F(p->r) + F(p->l)); }
```

8. Напишите функцию (или алгоритм) для добавления нового элемента в конец двусвязного нециклического списка. Элемент списка хранит вещественное число.

ВАРИАНТ 32

1. Вычислить значение переменной a

```
int a; a = 'a' != ('a' % 'a') << 'a' + 'a'/'a';
```

2. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = b; d = &b[1] + 1; a = *d - b[1] + *(++p);
```

3. Вычислить значение переменной a

```
int a; if (!(0 << 0 | !0) >= 0) a = 1; else a = 0;
```

4. Подставьте формальный параметр функции:

```
int a[3]={1,2,3}; int x; x=f(&a[0]);… int f(…){…}
```

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
int func(int a, int b){if(a<b)return(b);else
return(a);}
void main(){int m=10, n=5, k; k = func(m,n)%2+n/2;}
```

6. Написать функцию f, заменяющую цифры в символьной строке на точки.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int k; int v[10]; A *l,*r; };
int F(A *q){ int i,n,m;if (q==NULL) return 0;
for (n=0,i=0; i<k; i++) n+=q->v[i]);
return n+F(q->l)+F(q->r); }
```

8. Напишите функцию (или алгоритм) для добавления нового элемента в начало односвязного нециклического списка. Элемент списка хранит вещественное число.

ВАРИАНТ 33

1. Вычислить значение переменной a

```
int a; a = 13 - 13 % 13 + !((13 == 13) != 13);
```

2. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b + b[1]; d = b; a = (*p + 1)/(*d);
```

3. Вычислить значение переменной a

```
int a; if ('z' - 'a' > 'a' - 'z') a = 1; else a = 0;
```

4. Подставьте формальный параметр функции:

```
double z; char t='t'; z=f(&t); … double f(…){…}
```

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
int func(int a, int b){int c; c=a/b; return(c);}
void main(){int m=7, n=5, k; k = func(m,n)+3/2;}
```

6. Написать функцию f, находящую сумму отрицательных элементов массива вещественных чисел.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int val; A *next,*pred; };
A *F(A *p, int n) { for (; n!=0 && p!=NULL; n--,
p=p->next); return p; }
```

8. Напишите функцию (или алгоритм) для добавления нового элемента в конец односвязного нециклического списка. Элемент списка хранит целое число.

ВАРИАНТ 34

1. Вычислить значение переменной a

```
int a; a = !( 6 ? 7 - (!0 <= '0') + !5 : '5');
```

2. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = 2 + b; d = 5 + b; a = *d/(*p++);
```

3. Вычислить значение переменной a

```
int a;
if (1 + 2 + 3 + 4 + 'a' >> 0) a = 1; else a = 0;
```

4. Сколько раз выполняются операторы цикла

```
int k=3; while(1) {if(k=1)break; else {k--; }}
```

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
int func(int a, int b){if(a>=b)return(a);else
return(b);}
void main(){int m=10, n=5, k; k = func(m,n)%2;}
```

6. Написать функцию f, сравнивающую длины двух символьных строк.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int val; A *next; };
A *F(A *ph, int v)
{ struct A *q; q = new A; q->val = v; q->next = ph;
ph = q; return ph; }
```

8. Напишите функцию (или алгоритм) для добавления нового элемента в начало двусвязного нециклического списка. Элемент списка хранит строку символов.

ВАРИАНТ 35

1. Вычислить значение переменной a

```
int a; a = ((5 - '5'/'6') | !(33 + '3' + '3'));
```

2. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b + 2; d = &(*(p + 1)); a = *p + *d++;
```

3. Вычислить значение переменной a `int a;`

```
if (1 * 2 * 3 + 1 + 2 + 3 != 1) a = 1; else a = 0;
```

4. Можно ли использовать для разветвления такой оператор:

```
if(k=-3){…}; else {…};
```

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
int func(int a, int b){if(a>=b)return(a);else
return(b);}
void main(){int m=10, n=5, k; k = func(m,n)%2;}
```

6. Написать функцию f, вычисляющую факториал заданного целого числа.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int v;  A *l,*r; };
int F( A *p)
{    if (p==NULL) return(0);
return (1 + F(p->r) + F(p->l)); }
```

8. Напишите функцию (или алгоритм) для добавления нового элемента в конец двусвязного нециклического списка. Элемент списка хранит вещественное число.

ВАРИАНТ 36

1. Вычислить значение переменной a

```
int a;  a = (sizeof (char) || ('t' - 1)) << (2 & 3);
```

2. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = b + '2' - '1'; d = p++; a = *p - 'u';
```

3. Вычислить значение переменной a

```
int a; if (1 >> 2 ? 6 : !(13)) a = 1; else a = 0;
```

4. Для обращения к одному и тому же элементу массива `int a[3]={1,2,3};` предложены три варианта, укажите верные: 1)a[1]; 2) *(a+1); 3) *(&a[0]+1)

5. Вычислите значение переменной k. Содержательно сформулируйте, что делает функция.

```
double func(double a){double c; c=1.0/a; return(c);}
void main(){double m=10, n=2, k; k = func(m/2)+2/4;}
```

6. Написать функцию f, заменяющую в символьном массиве строчные латинские буквы на прописные.

7. Определите тип структуры данных. Содержательно сформулируйте, что делает функция F

```
struct A { int val; A *next,*pred; };
int  F(A *p){  int n;
for (n=0; p!=NULL; p=p->next, n++);
return(n); }
```

8. Напишите функцию (или алгоритм) для добавления нового элемента в начало односвязного нециклического списка. Элемент списка хранит строку символов.

ВАРИАНТ 37

1. Вычислить значение переменной a

```
int a;        a = (24 >> 2 << 3) - ('A' + 1) % 'A';
```

2. Вычислить значение переменной a

```
int a, b[5] = {2, 1, 3, '4' - '0', 5}, i = 2;
a = (int)(3/2.0) - b[i] + i++;
```

3. Вычислить значение переменной a

```
char s[] = "don't worry be happy", a; int i = 7;
a = (s[++i] > 0) && s[i + 1];
```

4. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b; d = &b[*p++]; a = 15/(*p);
```

5. Вычислить значение переменной a

```
int a; if ('n' / 'g' % 't' * 'u') a = 1; else a = 0;
```

6. Определить тип описателя v

```
double ** (*v) (double *a, int b);
```

7. Определить тип описателя v

```
typedef double P (char *a); P *v[5];
```

8. Написать функцию f, возвращающую указатель на последний элемент символьной строки.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – массив указателей на символьные строки.

ВАРИАНТ 38

1. Вычислить значение переменной a

```
int a;
a = (15 && 10 & 5) ? sizeof (char) : '9' - '0' << 2;
```

2. Вычислить значение переменной a

```
int a, b[5] = {0, 0, 3 < 2, 11, 5}, i = 1;
a = (int)(3/2.0) - b[i] + i++;
```

3. Вычислить значение переменной a

```
char s[] = "mein herz brennt", a; int i = 8;
a = s[i++] - s[i] - 0 ? -1 : 99;
```

4. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = b; d = &b[0]; a = *p >= *d++;
```

5. Вычислить значение переменной a

```
int a; if (11 || 12 | (13 && 14)) a = 1; else a = 0;
```

6. Определить тип описателя v

```
char *v (void (*a) (int b), int c);
```

7. Определить тип описателя v

```
typedef int (**P) (); P *v;
```

8. Написать функцию f, возвращающую указатель на заданный элемент массива.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – массив указателей на функции (функции определить самостоятельно).

ВАРИАНТ 39

1. Вычислить значение переменной a

```
int a;        a = 36 / 12 % 2 > 's' < 1 << 3;
```

2. Вычислить значение переменной a

```
int a, b[5] = {2, 'a' == 'a', 3, 4, 5}, i = 3;
a = -b[i] + (i++) - (int)(5.0/2.0);
```

3. Вычислить значение переменной a

```
char s[] = "informatika", a; int i = 2;
a = (s[i + 1] > 0) && (s[i + 1] > '\0');
```

4. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b + 4; d = &b[2]; a = *d + *--p;
```

5. Вычислить значение переменной a

```
int a; if (1 >> 2 > 3 << 4 < 5) a = 1; else a = 0;
```

6. Определить тип описателя v

```
char **v[33];
```

7. Определить тип описателя v

```
typedef char **P (char (*a) (), int b); P v;
```

8. Написать функцию f для подсчета ненулевых элементов в массиве вещественных чисел.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – указатель на функцию, возвращающую указатель на массив чисел.

ВАРИАНТ 40

1. Вычислить значение переменной a

```
int a;        a = 5 * 6 & 10 != 5 << 1;
```

2. Вычислить значение переменной a

```
int a, b[5] = {0, 10, 3 << 2, 1, 1}, i = 1;
a = (int)(0/2.0) - b[i] + i++;
```

3. Вычислить значение переменной a

```
char s[] = "press any key", a; int i = 0;
a = s[i++] - s[i] - 0 ? -1 : 99;
```

4. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = &(*(b + 1)); d = ++p; a = *p - *d;
```

5. Вычислить значение переменной a

```
int a; if (!(1 == !(1 || 1))) a = 1; else a = 0;
```

6. Определить тип описателя v

```
int (*(*v) ()) [5];
```

7. Определить тип описателя v

```
typedef int P[10]; P *v (int a);
```

8. Написать функцию f для формирования массива простых чисел, не больших заданного.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – указатель на матрицу указателей на строки.

ВАРИАНТ 41

1. Вычислить значение переменной a

```
int a;        a = 'A' - 'D' + '7' % '5' < (16 >> 2);
```

2. Вычислить значение переменной a

```
int a, b[5] = {2, 1, 3, '4' - '0', 5}, i = 2;
a = b[i + 1] - b[i++] > 1;
```

3. Вычислить значение переменной a

```
char s[] = "the matrix has you", a; int i = 4;
a = (s[i] + 1)%(s[i++] - 1);
```

4. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b; d = &b[b[1]]; a = *p + *d++;
```

5. Вычислить значение переменной a

```
int a; if (16 - '6' < 1 << 2) a = 1; else a = 0;
```

6. Определить тип описателя v

```
float (*v (int a)) [5];
```

7. Определить тип описателя v

```
typedef double (*(*P) (char a[])) (); P v;
```

8. Написать функцию f, возвращающую указатель на минимальный элемент символьной строки.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – массив переменных структурного типа (определить самостоятельно).

ВАРИАНТ 42

1. Вычислить значение переменной a

```
int a;
a = !0 ? 15 : (13 << 1 ? '8' : 2 * sizeof (char));
```

2. Вычислить значение переменной a

```
int a, b[5] = {0, 0, 3 < 2, 11, 5}, i = 1;
a = b[i + 1] - b[i/2] >> 1 - i--;
```

3. Вычислить значение переменной a

```
char s[] = "123123123", a; int i = 3;
a = (s[3] - s['3' - '0']) + s[i++];
```

4. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = b; d = &b[1] + 1; a = *d - b[1] + *(++p);
```

5. Вычислить значение переменной a

```
int a; if ((2 != 22)*1 == 1) a = 1; else a = 0;
```

6. Определить тип описателя v

```
double *v[2][10];
```

7. Определить тип описателя v

```
typedef double P (double a, double b); P **v;
```

8. Написать функцию f, заменяющую цифры в символьной строке на точки.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – указатель на функцию, возвращающую указатель на строку.

ВАРИАНТ 43

1. Вычислить значение переменной a

```
int a; a = 'a' > 'b' < 'c' - 52 >> 1 % ('e' - 'a');
```

2. Вычислить значение переменной a

```
int a, b[5] = {2, 'a' == 'a', 3, 4, 5}, i = 3;
a = b[i + 1] % b[i++] < 0;
```

3. Вычислить значение переменной a

```
char s[] = "informatika", a; int i = 0;
a = (s[i] + 1)%(s[i++] - 1);
```

4. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b + b[1]; d = b; a = (*p + 1)/(*d);
```

5. Вычислить значение переменной a

```
int a; if (sizeof (char)%6 == 6) a = 1; else a = 0;
```

6. Определить тип описателя v

```
void (*v) (char a[]);
```

7. Определить тип описателя v

```
typedef int **P; P *v[10];
```

8. Написать функцию f, находящую сумму отрицательных элементов массива вещественных чисел.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – массив указателей на матрицы целых чисел.

ВАРИАНТ 44

1. Вычислить значение переменной a

```
int a;    a = (15 == 0) || (15 != '0' - '5') / 3;
```

2. Вычислить значение переменной a

```
int a, b[5] = {0, 10, 3 << 2, 1, 1}, i = 1;
a = b[i + 1] - b[i/2] >> 1 - i--;
```

3. Вычислить значение переменной a

```
char s[] = "aller anfang ist schwer", a; int i = 6;
a = (s[3] - s['3' - '0']) + s[i++];
```

4. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = 2 + b; d = 5 + b; a = *d/(*p++);
```

5. Вычислить значение переменной a

```
int a; if (11 | 11 || 11 == 11) a = 1; else a = 0;
```

6. Определить тип описателя v

```
int **v (int **a);
```

7. Определить тип описателя v

```
typedef int (*(*P) ()) [5]; P v;
```

8. Написать функцию f, сравнивающую длины двух символьных строк.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – структурная переменная, содержащая массив строк.

ВАРИАНТ 45

1. Вычислить значение переменной a

```
int a; a = '7' - '5' + (2 << sizeof (char)) % 4 / 2;
```

2. Вычислить значение переменной a

```
int a, b[5] = {2, 1, 3, '4' - '0', 5}, i = 2;
a = 16 % (1 + b[--i]) + b[i - 1];
```

3. Вычислить значение переменной a

```
char s[] = "don't worry be happy", a; int i = 7;
a = (s[i] + 1)%(s[i++] - 1);
```

4. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b + 2; d = &(*(p + 1)); a = *p + *d++;
```

5. Вычислить значение переменной a

```
int a; if (2 & 2 | 2 && 2) a = 1; else a = 0;
```

6. Определить тип описателя v

```
char *(*v[8]) (char *a);
```

7. Определить тип описателя v

```
typedef int (*P (unsigned int **a)) [5]; P v;
```

8. Написать функцию f, вычисляющую факториал заданного целого числа.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – массив указателей на массивы указателей на символы.

ВАРИАНТ 46

1. Вычислить значение переменной a

```
int a; a = sizeof (char) || ('3' - 1) <= (22 || 3);
```

2. Вычислить значение переменной a

```
int a, b[5] = {0, 0, 3 < 2, 11, 5}, i = 1;
a = 16 % (1 + b[i++]) + b[i - 1|1];
```

3. Вычислить значение переменной a

```
char s[] = "mein herz brennt", a; int i = 8;
a = (s['3'-'3'] - s['5'-'5']) + s[i++];
```

4. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = b + '2' - '1'; d = p++; a = *p - 'u';
```

5. Вычислить значение переменной a

```
int a; if (0 << !0 < 1 >> !1) a = 1; else a = 0;
```

6. Определить тип описателя v

```
char *(*v) (int a, int b);
```

7. Определить тип описателя v

```
typedef int *P[5]; P *v[5];
```

8. Написать функцию f, заменяющую в символьном массиве строчные латинские буквы на прописные.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – массив указателей на символьные строки.

ВАРИАНТ 47

1. Вычислить значение переменной a

```
int a;  a = (24 << 2 < 3) - ('A' + 1) % 'B';
```

2. Вычислить значение переменной a

```
int a, b[5] = {2, 'a' == 'a', 3, 4, 5}, i = 3;
a = 11 % (11 + b[--i]) + b[i + 1];
```

3. Вычислить значение переменной a

```
char s[] = "wanted dead or alive", a; int i = 9;
a = (s[i] + 1)%(s[i++] - 1);
```

4. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b; d = &b[*p++]; a = 15/(*p);
```

5. Вычислить значение переменной a

```
int a; if (25/(3 | 2)%2 > 0) a = 1; else a = 0;
```

6. Определить тип описателя v

```
int *v (int a, int b[]);
```

7. Определить тип описателя v

```
typedef char (*P) (char *a); P v;
```

8. Написать функцию f, возвращающую указатель на последний элемент символьной строки.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – массив указателей на функции (функции определить самостоятельно).

ВАРИАНТ 48

1. Вычислить значение переменной a

```
int a;
a = (15 & 10 >= 5) ? sizeof (char) : 9 - 0 << 2;
```

2. Вычислить значение переменной a

```
int a, b[5] = {0, 10, 3 << 2, 1, 1}, i = 1;
a = (16 % 11) + b[i++] + b[1];
```

3. Вычислить значение переменной a

```
char s[] = "press any key", a; int i = 0;
a = (s['3' - '3'] - s[6 - 5]) + s[i++];
```

4. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = b; d = &b[0]; a = *p >= *d++;
```

5. Вычислить значение переменной a

```
int a; if (15 - 16 ? 15 == 16 : 15 != 16) a = 1;
else a = 0;
```

6. Определить тип описателя v

```
void (*v[5]) ();
```

7. Определить тип описателя v

```
typedef (*P (int a)) [5]; P v;
```

8. Написать функцию f, возвращающую указатель на заданный элемент массива.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – указатель на функцию, возвращающую указатель на массив чисел.

ВАРИАНТ 49

1. Вычислить значение переменной a

```
int a;  a = 36 / 13 % 1 > 's' < sizeof (char) << 3;
```

2. Вычислить значение переменной a

```
int a, b[5] = {2, 1, 3, '4' - '0', 5}, i = 2;
a = b[(b[i + 1] - 2)] - b[i++];
```

3. Вычислить значение переменной a

```
char s[] = "the matrix has you", a; int i = 4;
a = s['5' - '3']++ - s[i + 1];
```

4. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b + 4; d = &b[2]; a = *d + *--p;
```

5. Вычислить значение переменной a

```
int a;
if ((int)(sizeof (char)) > -1) a = 1; else a = 0;
```

6. Определить тип описателя v

```
void (*v) (char **a);
```

7. Определить тип описателя v

```
typedef int (*P) (); P v[5];
```

8. Написать функцию f для подсчета ненулевых элементов в массиве вещественных чисел.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – указатель на матрицу указателей на строки.

ВАРИАНТ 50

1. Вычислить значение переменной a

```
int a;       a = 5 * 6 && 10 == 5 <= 1;
```

2. Вычислить значение переменной a

```
int a, b[5] = {0, 0, 3 < 2, 11, 5}, i = 1;
a = b[(b[i + 3] - 2)] - b[i++];
```

3. Вычислить значение переменной a

```
char s[] = "alarm partisanen!", a; int i = 3;
a = s[i] + s[i + 4] - s[i++];
```

4. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = &(*(b + 1)); d = ++p; a = *p - *d;
```

5. Вычислить значение переменной a

```
int a; if (13 + 13 >> 13 < 13) a = 1; else a = 0;
```

6. Определить тип описателя v

```
char (*v (float a)) (int b);
```

7. Определить тип описателя v

```
typedef void (*P) (char **a, int b); P v;
```

8. Написать функцию f для формирования массива простых чисел, не больших заданного.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – массив переменных структурного типа (определить самостоятельно).

ВАРИАНТ 51

1. Вычислить значение переменной a

```
int a;   a = 's' - 'p' - 17 % 15 < (16 >> 2);
```

2. Вычислить значение переменной a

```
int a, b[5] = {2, 'a' == 'a', 3, 4, 5}, i = 3;
a = b[(b[i + 1] - 2)] - b[i++];
```

3. Вычислить значение переменной a

```
char s[] = "qqq wwww eeeee rrrrrr", a; int i = 10;
a = s['5' - '3']++ - s[i + 1];
```

4. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b; d = &b[b[1]]; a = *p + *d++;
```

5. Вычислить значение переменной a

```
int a; if (55 | 54 | 53 || 1) a = 1; else a = 0;
```

6. Определить тип описателя v

```
int *(*v[10]) [5];
```

7. Определить тип описателя v

```
typedef void (*P ()) ();P *v;
```

8. Написать функцию f, возвращающую указатель на минимальный элемент символьной строки.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – указатель на функцию, возвращающую указатель на строку.

ВАРИАНТ 52

1. Вычислить значение переменной a

```
int a;  a = (10 == '10') ? 15 : (13 << 1 ? 8 : 2);
```

2. Вычислить значение переменной a

```
int a, b[5] = {0, 10, 3 << 2, 1, 1}, i = 1;
a = b[(b[i + 3] - 1)] - b[i++];
```

3. Вычислить значение переменной a

```
char s[] = "aller anfang ist schwer", a; int i = 6;
a = s[i] + s[i + 3] - s[i++];
```

4. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = b; d = &b[1] + 1; a = *d - b[1] + *(++p);
```

5. Вычислить значение переменной a

```
int a; if (!(0 << 0 | !0) >= 0) a = 1; else a = 0;
```

6. Определить тип описателя v

```
double ** (*v) (double *a, int b);
```

7. Определить тип описателя v

```
typedef double P (char *a); P *v[5];
```

8. Написать функцию f, заменяющую цифры в символьной строке на точки.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – массив указателей на матрицы целых чисел.

ВАРИАНТ 53

1. Вычислить значение переменной a

```
int a;     a = 1 > 2 >= 'c' - 2 << 1 / ('c' - 'a');
```

2. Вычислить значение переменной a

```
int a, b[5] = {2, 1, 3, '4' - '0', 5}, i = 2;
a = b[i++]/b[i - 1]%2 + 't' - 't';
```

3. Вычислить значение переменной a

```
char s[] = "don't worry be happy", a; int i = 7;
a = s['5' - '3']++ - s[i++];
```

4. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b + b[1]; d = b; a = (*p + 1)/(*d);
```

5. Вычислить значение переменной a

```
int a; if ('z' - 'a' > 'a' - 'z') a = 1; else a = 0;
```

6. Определить тип описателя v

```
char *v (void (*a) (int b), int c);
```

7. Определить тип описателя v

```
typedef int (**P) (); P *v;
```

8. Написать функцию f, находящую сумму отрицательных элементов массива вещественных чисел.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – структурная переменная, содержащая массив строк.

ВАРИАНТ 54

1. Вычислить значение переменной a

```
int a;    a = ('b' == 'n') && (15 <= 0 - '5') % 3;
```

2. Вычислить значение переменной a

```
int a, b[5] = {0, 0, 3 < 2, 11, 5}, i = 1;
a = b[i++]/b[i + 2]%2 - '1' + '1';
```

3. Вычислить значение переменной a

```
char s[] = "mein herz brennt", a; int i = 8;
a = s[i] + s[i + 3] - s[i++];
```

4. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = 2 + b; d = 5 + b; a = *d/(*p++);
```

5. Вычислить значение переменной a

```
int a;
if (1 + 2 + 3 + 4 + 'a' >> 0) a = 1; else a = 0;
```

6. Определить тип описателя v

```
char **v[33];
```

7. Определить тип описателя v

```
typedef char **P (char (*a) (), int b); P v;
```

8. Написать функцию f, сравнивающую длины двух символьных строк.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – массив указателей на массивы указателей на символы.

ВАРИАНТ 55

1. Вычислить значение переменной a

```
int a;     a = '5' - '7' + (2 << '2' - '1') / 4 % 2;
```

2. Вычислить значение переменной a

```
int a, b[5] = {2, 'a' == 'a', 3, 4, 5}, i = 3;
a = b[i++]%b[i - 1]%2 - '0' + '6';
```

3. Вычислить значение переменной a

```
char s[] = "what do you want?", a; int i = 9;
a = s[55 - 52]++ - s[i++];
```

4. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b + 2; d = &(*(p + 1)); a = *p + *d++;
```

5. Вычислить значение переменной a

```
int a;
if (1 * 2 * 3 + 1 + 2 + 3 != 1) a = 1; else a = 0;
```

6. Определить тип описателя v

```
int (*(*v) ()) [5];
```

7. Определить тип описателя v

```
typedef int P[10]; P *v (int a);
```

8. Написать функцию f, вычисляющую факториал заданного целого числа.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – массив указателей на символьные строки.

ВАРИАНТ 56

1. Вычислить значение переменной a

```
int a;    a = (sizeof (char) & (1 != 1)) << (2 & 3);
```

2. Вычислить значение переменной a

```
int a, b[5] = {0, 10, 3 << 2, 1, 1}, i = 1;
a = b[i++]/b[i + 2]%2 + 't' - 't';
```

3. Вычислить значение переменной a

```
char s[] = "press any key", a; int i = 0;
a = s[i++] + s[i + 2] - s[i];
```

4. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = b + '2' - '1'; d = p++; a = *p - 'u';
```

5. Вычислить значение переменной a

```
int a; if (1 >> 2 ? 6 : !(13)) a = 1; else a = 0;
```

6. Определить тип описателя v

```
float (*v (int a)) [5];
```

7. Определить тип описателя v

```
typedef double (*(*P) (char a[])) (); P v;
```

8. Написать функцию f, заменяющую в символьном массиве строчные латинские буквы на прописные.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – массив указателей на функции (функции определить самостоятельно).

ВАРИАНТ 57

1. Вычислить значение переменной a

```
int a;    a = (24 && 2 & 3) + ('B' - 1) % 'A';
```

2. Вычислить значение переменной a

```
int a, b[5] = {2, 1, 3, '4' - '0', 5}, i = 2;
a = (int)(3/2.0) - b[i] + i++;
```

3. Вычислить значение переменной a

```
char s[] = "the matrix has you", a; int i = 4;
a = s[++i] + s[1 + i] - s[i + 2];
```

4. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b; d = &b[*p++]; a = 15/(*p);
```

5. Вычислить значение переменной a

```
int a; if ('n' / 'g' % 't' * 'u') a = 1; else a = 0;
```

6. Определить тип описателя v

```
double *v[2][10];
```

7. Определить тип описателя v

```
typedef double P (double a, double b); P **v;
```

8. Написать функцию f, возвращающую указатель на последний элемент символьной строки.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – указатель на функцию, возвращающую указатель на массив чисел.

ВАРИАНТ 58

1. Вычислить значение переменной a

```
int a;
a = (55 - 54 - 53) ? sizeof (char)*2 : 2 - (0 << 2);
```

2. Вычислить значение переменной a

```
int a, b[5] = {0, 0, 3 < 2, 11, 5}, i = 1;
a = (int)(3/2.0) - b[i] + i++;
```

3. Вычислить значение переменной a

```
char s[] = "alarm partisanen!", a; int i = 3;
a = s['n' - s[++i]] - s[i];
```

4. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = b; d = &b[0]; a = *p >= *d++;
```

5. Вычислить значение переменной a

```
int a; if (11 || 12 | (13 && 14)) a = 1; else a = 0;
```

6. Определить тип описателя v

```
void (*v) (char a[]);
```

7. Определить тип описателя v

```
typedef int **P; P *v[10];
```

8. Написать функцию f, возвращающую указатель на заданный элемент массива.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – указатель на матрицу указателей на строки.

ВАРИАНТ 59

1. Вычислить значение переменной a

```
int a; a = 2 / 2 % 2 >= '2' < 2 << 2;
```

2. Вычислить значение переменной a

```
int a, b[5] = {2, 'a' == 'a', 3, 4, 5}, i = 3;
a = -b[i] + (i++) - (int)(5.0/2.0);
```

3. Вычислить значение переменной a

```
char s[] = "012345678901234567890", a; int i = 10;
a = s[--i] + s[1 + i] - s[i + 2];
```

4. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b + 4; d = &b[2]; a = *d + *--p;
```

5. Вычислить значение переменной a

```
int a; if (1 >> 2 > 3 << 4 < 5) a = 1; else a = 0;
```

6. Определить тип описателя v

```
int **v (int **a);
```

7. Определить тип описателя v

```
typedef int (*(*P) ()) [5]; P v;
```

8. Написать функцию f для подсчета ненулевых элементов в массиве вещественных чисел.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – массив переменных структурного типа (определить самостоятельно).

ВАРИАНТ 60

1. Вычислить значение переменной a

```
int a; a = 5 - 4 && 10 != (5 | 5) << 1;
```

2. Вычислить значение переменной a

```
int a, b[5] = {0, 10, 3 << 2, 1, 1}, i = 1;
a = (int)(0/2.0) - b[i] + i++;
```

3. Вычислить значение переменной a

```
char s[] = "aller anfang ist schwer", a; int i = 6;
a = s['n' - s[++i]] - s[i - 1];
```

4. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = &(*(b + 1)); d = ++p; a = *p - *d;
```

5. Вычислить значение переменной a

```
int a; if (!(1 == !(1 || 1))) a = 1; else a = 0;
```

6. Определить тип описателя v

```
char *(*v[8]) (char *a);
```

7. Определить тип описателя v

```
typedef int (*P (unsigned int **a)) [5]; P v;
```

8. Написать функцию f для формирования массива простых чисел, не больших заданного.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – указатель на функцию, возвращающую указатель на строку.

ВАРИАНТ 61

1. Вычислить значение переменной a

```
int a; a = !0 + 6 + '7' / '5' << 1 > 2;
```

2. Вычислить значение переменной a

```
int a, b[5] = {2, 1, 3, '4' - '0', 5}, i = 2;
a = b[i + 1] - b[i++] > 1;
```

3. Вычислить значение переменной a

```
char s[] = "don't worry be happy", a; int i = 7;
a = s[--i] + s[i + 3] - s[i + 2];
```

4. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b; d = &b[b[1]]; a = *p + *d++;
```

5. Вычислить значение переменной a

```
int a; if (16 - '6' < 1 << 2) a = 1; else a = 0;
```

6. Определить тип описателя v

```
char *(*v) (int a, int b);
```

7. Определить тип описателя v

```
typedef int *P[5]; P *v[5];
```

8. Написать функцию f, возвращающую указатель на минимальный элемент символьной строки.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – массив указателей на матрицы целых чисел.

ВАРИАНТ 62

1. Вычислить значение переменной a

```
int a;
a = 10 ? 10 : (10 << sizeof (char) ? 10 : 10 * 10);
```

2. Вычислить значение переменной a

```
int a, b[5] = {0, 0, 3 < 2, 11, 5}, i = 1;
a = b[i + 1] - b[i/2] >> 1 - i--;
```

3. Вычислить значение переменной a

```
char s[] = "mein herz brennt", a; int i = 8;
a = s['n' - s[3]] - s[i - 5];
```

4. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = b; d = &b[1] + 1; a = *d - b[1] + *(++p);
```

5. Вычислить значение переменной a

```
int a; if ((2 != 22)*1 == 1) a = 1; else a = 0;
```

6. Определить тип описателя v

```
int *v (int a, int b[]);
```

7. Определить тип описателя v

```
typedef char (*P) (char *a); P v;
```

8. Написать функцию f, заменяющую цифры в символьной строке на точки.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – структурная переменная, содержащая массив строк.

ВАРИАНТ 63

1. Вычислить значение переменной a

```
int a; a = 'a' < 'b' < 'c' - 'c' >> 1 % ('b' - 'a');
```

2. Вычислить значение переменной a

```
int a, b[5] = {2, 'a' == 'a', 3, 4, 5}, i = 3;
a = b[i + 1] % b[i++] < 0;
```

3. Вычислить значение переменной a

```
char s[] = "what do you want?", a; int i = 9;
a = s[--i] - s[i] + s[i + 2];
```

4. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b + b[1]; d = b; a = (*p + 1)/(*d);
```

5. Вычислить значение переменной a

```
int a; if (sizeof (char)%6 == 6) a = 1; else a = 0;
```

6. Определить тип описателя v

```
void (*v[5]) ();
```

7. Определить тип описателя v

```
typedef (*P (int a)) [5]; P v;
```

8. Написать функцию f, находящую сумму отрицательных элементов массива вещественных чисел.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – массив указателей на массивы указателей на символы.

ВАРИАНТ 64

1. Вычислить значение переменной a

```
int a; a = (15 >= 0) | (15 != 0 - 5) + 3;
```

2. Вычислить значение переменной a

```
int a, b[5] = {0, 10, 3 << 2, 1, 1}, i = 1;
a = b[i + 1] - b[i/2] >> 1 - i--;
```

3. Вычислить значение переменной a

```
char s[] = "press any key", a; int i = 0;
a = s['t' - s[3]] - s[i + 3];
```

4. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = 2 + b; d = 5 + b; a = *d/(*p++);
```

5. Вычислить значение переменной a

```
int a; if (11 | 11 || 11 == 11) a = 1; else a = 0;
```

6. Определить тип описателя v

```
void (*v) (char **a);
```

7. Определить тип описателя v

```
typedef int (*P) (); P v[5];
```

8. Написать функцию f, сравнивающую длины двух символьных строк.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – массив указателей на символьные строки.

ВАРИАНТ 65

1. Вычислить значение переменной a

```
int a; a = 7 - 77 + (7 << sizeof (char)) / 77 % 7;
```

2. Вычислить значение переменной a

```
int a, b[5] = {2, 1, 3, '4' - '0', 5}, i = 2;
a = 16 % (1 + b[--i]) + b[i - 1];
```

3. Вычислить значение переменной a

```
char s[] = "the matrix has you", a; int i = 4;
a = s[s[i++] - 'm']/'t';
```

4. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b + 2; d = &(*(p + 1)); a = *p + *d++;
```

5. Вычислить значение переменной a

```
int a; if (2 & 2 | 2 && 2) a = 1; else a = 0;
```

6. Определить тип описателя v

```
char (*v (float a)) (int b);
```

7. Определить тип описателя v

```
typedef void (*P) (char **a, int b); P v;
```

8. Написать функцию f, вычисляющую факториал заданного целого числа.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – массив указателей на функции (функции определить самостоятельно).

ВАРИАНТ 66

1. Вычислить значение переменной a

```
int a; a = !0 > 0 <= 0 + !0 << 0;
```

2. Вычислить значение переменной a

```
int a, b[5] = {0, 0, 3 < 2, 11, 5}, i = 1;
a = 16 % (1 + b[i++]) + b[i - 1|1];
```

3. Вычислить значение переменной a

```
char s[] = "alarm partisanen!", a; int i = 3;
a = ('5' > '4') + s[++i];
```

4. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = b + '2' - '1'; d = p++; a = *p - 'u';
```

5. Вычислить значение переменной a

```
int a; if (0 << !0 < 1 >> !1) a = 1; else a = 0;
```

6. Определить тип описателя v

```
int *(*v[10]) [5];
```

7. Определить тип описателя v

```
typedef void (*P ()) ();P *v;
```

8. Написать функцию f, заменяющую в символьном массиве строчные латинские буквы на прописные.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – указатель на функцию, возвращающую указатель на массив чисел.

ВАРИАНТ 67

1. Вычислить значение переменной a

```
int a; a = 'a' != ('a' % 'a') << 'a' + 'a'/'a';
```

2. Вычислить значение переменной a

```
int a, b[5] = {2, 'a' == 'a', 3, 4, 5}, i = 3;
a = 11 % (11 + b[--i]) + b[i + 1];
```

3. Вычислить значение переменной a

```
char s[] = "qwerty01234567890", a; int i = 10;
a = s['6' - s[--i]]%'r';
```

4. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b; d = &b[*p++]; a = 15/(*p);
```

5. Вычислить значение переменной a

```
int a; if (25/(3 | 2)%2 > 0) a = 1; else a = 0;
```

6. Определить тип описателя v

```
double ** (*v) (double *a, int b);
```

7. Определить тип описателя v

```
typedef double P (char *a); P *v[5];
```

8. Написать функцию f, возвращающую указатель на последний элемент символьной строки.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – указатель на матрицу указателей на строки.

ВАРИАНТ 68

1. Вычислить значение переменной a

```
int a; a = 13 - 13 % 13 + !((13 == 13) != 13);
```

2. Вычислить значение переменной a

```
int a, b[5] = {0, 10, 3 << 2, 1, 1}, i = 1;
a = (16 % 11) + b[i++] + b[1];
```

3. Вычислить значение переменной a

```
char s[] = "aller anfang ist schwer", a; int i = 6;
a = ('5' > '4') + s[++i];
```

4. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = b; d = &b[0]; a = *p >= *d++;
```

5. Вычислить значение переменной a

```
int a; if (15 - 16 ? 15 == 16 : 15 != 16) a = 1;
else a = 0;
```

6. Определить тип описателя v

```
char *v (void (*a) (int b), int c);
```

7. Определить тип описателя v

```
typedef int (**P) (); P *v;
```

8. Написать функцию f, возвращающую указатель на заданный элемент массива.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – массив переменных структурного типа (определить самостоятельно).

ВАРИАНТ 69

1. Вычислить значение переменной a

```
int a; a = !( 6 ? 7 - (!0 <= '0') + !5 : '5');
```

2. Вычислить значение переменной a

```
int a, b[5] = {2, 1, 3, '4' - '0', 5}, i = 2;
a = b[(b[i + 1] - 2)] - b[i++];
```

3. Вычислить значение переменной a

```
char s[] = "don't worry be happy", a; int i = 7;
a = s['x' - s[--i]]/'r';
```

4. Вычислить значение переменной a

```
int b[5] = {1, 2, 3, 4, 5}, a, *p, *d;
p = b + 4; d = &b[2]; a = *d + *--p;
```

5. Вычислить значение переменной a

```
int a;
if ((int)(sizeof (char)) > -1) a = 1; else a = 0;
```

6. Определить тип описателя v

```
char **v[33];
```

7. Определить тип описателя v

```
typedef char **P (char (*a) (), int b); P v;
```

8. Написать функцию f для подсчета ненулевых элементов в массиве вещественных чисел.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – указатель на функцию, возвращающую указатель на строку.

ВАРИАНТ 70

1. Вычислить значение переменной a

```
int a; a = ((5 - '5'/'6') | !(33 + '3' + '3'));
```

2. Вычислить значение переменной a

```
int a, b[5] = {0, 0, 3 < 2, 11, 5}, i = 1;
a = b[(b[i + 3] - 2)] - b[i++];
```

3. Вычислить значение переменной a

```
char s[] = "mein herz brennt", a; int i = 8;
a = ('8' >= '9') + s[++i];
```

4. Вычислить значение переменной a

```
char b[] = "Astral Dominae", a, *p, *d;
p = &(*(b + 1)); d = ++p; a = *p - *d;
```

5. Вычислить значение переменной a

```
int a; if (13 + 13 >> 13 < 13) a = 1; else a = 0;
```

6. Определить тип описателя v

```
int (*(*v) ()) [5];
```

7. Определить тип описателя v

```
typedef int P[10]; P *v (int a);
```

8. Написать функцию f для формирования массива простых чисел, не больших заданного.

9. Привести пример вызова функции f из п. 8 с помощью указателя на нее.

10. Выделить динамически память для хранения объекта и инициализировать его. Объект – массив указателей на матрицы целых чисел.