Министерство образования и науки Российской Федерации

федеральное государственное бюджетное образовательное учреждение высшего образования «Казанский национальный исследовательский технический университет им. А.Н. Туполева-КАИ»

Институт (факультет) **Институт компьютерных технологий и защиты информации**

Кафедра **Автоматизированных систем обработки информации и управления**

УТВЕРЖДАЮ
Ответственный за ОП
_____ М.П. Шлеймович
«31» 08 2017 г.
Регистрационный номер 4030.17.10

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

для проведения промежуточной аттестации обучающихся по дисциплине (модулю) или практике

«Основы информатики и программирования»

(наименование дисциплины, практики)

Индекс по учебному плану: **Б1.Б.11.01.**

Направление подготовки: **09.03.02 «Информационные системы и технологии».**

Квалификация: **бакалавр.**

Профиль подготовки: **«Информационные системы».**

Вид(ы) профессиональной деятельности: **научно-исследовательская, производственно-технологическая.**

Заведующий кафедрой АСОИУ: М.П. Шлеймович

Разработчик: доцент кафедры АСОИУ Д.Г.Хохлов

Казань 2017 г.

Фонд оценочных средств для проведения промежуточной аттестации обучающихся по дисциплине (модулю) или практике

«Основы информатики и программирования»

(наименование дисциплины, практики)

Содержание фонда оценочных средств (ФОС) соответствует требованиям федерального государственного стандарта высшего образования (ФГОС ВО) по направлению подготовки 09.03.02 «Информационные системы и технологии», учебному плану направления 09.03.02 «Информационные системы и технологии».

Разработанные ФОС обладают необходимой полнотой и являются актуальными для оценки компетенций, осваиваемых обучающимися при изучении дисциплины. Они полностью соответствуют задачам будущей профессиональной деятельности обучающихся, установленных ФГОС ВО. В составе ФОС имеются оценочные средства в виде тестовых заданий и контрольных вопросов различного уровня сложности, которые позволяют провести оценку порогового, продвинутого и превосходного уровней освоения компетенций по дисциплине.

ФОС обладают необходимой степенью приближенности к задачам будущей профессиональной деятельности обучающихся, связанным со способностью применять знания, умения и навыки для решения профессиональных задач, соответствующих компетенциям, реализуемым дисциплиной.

Замечания отсутствуют.

Заключение. Учебно-методическая комиссия делает вывод о том, что представленные материалы соответствуют требованиям ФГОС ВО по направлению подготовки 09.03.02 «Информационные системы и технологии» и рекомендуются для использования в учебном процессе.

Рассмотрено на заседании учебно-методической комиссии

«31» августа 2017 г., протокол № 8.

Председатель УМК _____ В.В. Родионов

Содержание

Введение

Фонд оценочных средств для проведения промежуточной аттестации обучающихся по дисциплине (ФОС ПА) «Основы информатики и программирования» – это комплект методических и контрольно-измерительных материалов, предназначенных для определения уровня сформированности компетенций, оценивания знаний, умений, владений на разных этапах освоения дисциплины для проведения промежуточной аттестации обучающихся по дисциплине.

ФОС ПА является составной частью учебного и методического обеспечения программы магистранта по направлению 09.03.02 «Информационные системы и технологии».

Задачи ФОС по дисциплине:

– оценка запланированных результатов освоения дисциплины (модуля) или практики обучающимися в процессе изучения дисциплины (модуля) или практики, в соответствии с разработанными и принятыми критериями по каждому виду контроля;

– контроль и управление процессом приобретения обучающимися необходимых знаний, умений, навыков и формирования компетенций, определенных в ФГОС ВО по направлению подготовки

ФОС ПА по дисциплине сформирован на основе следующих основных принципов оценивания:

– пригодности (валидности) (объекты оценки соответствуют поставленным целям обучения);

– надежности (использования единообразных стандартов и критериев для оценивания запланированных результатов);

– эффективности (соответствия результатов деятельности поставленным задачам).

ФОС ПА по дисциплине разработан в соответствии с требованиями ФГОС ВО для аттестации обучающихся на соответствие их персональных достижений требованиям поэтапного формирования соответствующих составляющих компетенций и включает контрольные вопросы (или тесты) и типовые задания, необходимые для оценки знаний, умений и навыков, характеризующих этапы формирования компетенций.

1 Формы промежуточной аттестации по дисциплине

Дисциплина «Основы информатики и программирования» изучается в 1 семестре на первом курсе при очной форме обучения и завершается промежуточной аттестацией в форме экзамена.

2 Оценочные средства для промежуточной аттестации

Оценочные средства для промежуточной аттестации по дисциплине.

Таблица 1

Оценочные средств для промежуточной аттестации
(очная форма обучения)

№ п/п	Семестр	Форма промежуточной аттестации	Оценочные средства
1.	1	экзамен	ФОС ПА

3 Перечень компетенций с указанием этапов их формирования в процессе освоения дисциплины

Перечень компетенций и их составляющих, которые должны быть сформированы при изучении темы соответствующего раздела дисциплины, представлен в таблице 2.

Таблица 2

Перечень компетенций и этапы их формирования
в процессе освоения дисциплины

№ п/п	Этап формирования (семестр)	Наименование раздела	Код формируемой компетенции (составляющей компетенции)		Форма промежуточной аттестации
1.	1	Основные понятия программирования	ОПК-1, ПК-17	ОПК-1З, ОПК-1У, ОПК-1В, ПК-17З, ПК-17У, ПК-17В	экзамен
2.	1	Технология программирования	ОПК-1, ПК-17	ОПК-1З, ОПК-1У, ОПК-1В, ПК-17З, ПК-17У, ПК-17В	экзамен

4 Описание показателей и критериев оценивания компетенций на различных этапах их формирования, описания шкалы оценивания

Показатели и критерии оценивания сформированности компетенций приведены в таблице 3.

Таблица 3

Показатели и критерии оценивания сформированности компетенций

№ п/п	Этап формирования (семестр)	Код формируемой компетенции (составляющей компетенции)		Критерии оценивания	Показатели оценивания (планируемые результаты обучения)		
					Пороговый уровень	Продвинутый уровень	Превосходный уровень
1.	3	ОПК-1	ОПК-1З	Теоретические навыки	Знать базовые методы информатики и инструменты оценки, выбора, использования и разработки программных средств решения простых практических задач в области информационных систем и технологий.	Знать базовые методы информатики и инструменты оценки, выбора, использования и разработки программных средств решения практических задач средней сложности в области информационных систем и технологий.	Знать базовые методы информатики и инструменты оценки, выбора, использования и разработки программных средств решения сложных практических задач в области информационных систем и технологий.
2.	3	ОПК-1	ОПК-1У ОПК-1В	Практические навыки	Уметь использовать базовые методы информатики и инструменты оценки, выбора, использования и разработки программных средств решения простых практических задач в области информационных систем и технологий.	Уметь использовать базовые методы информатики и инструменты оценки, выбора, использования и разработки программных средств решения практических задач средней сложности в области информационных систем и технологий.	Уметь использовать базовые методы информатики и инструменты оценки, выбора, использования и разработки программных средств решения сложных практических задач в области информационных систем и технологий.
					Владеть базовыми методами информатики и инструментами оценки, выбора, использования и разработки программных средств решения простых практических задач в области информационных систем и технологий.	Владеть базовыми методами информатики и инструментами оценки, выбора, использования и разработки программных средств решения практических задач средней сложности в области информационных систем и технологий.	Владеть базовыми методами информатики и инструментами оценки, выбора, использования и разработки программных средств решения сложных практических задач в области информационных систем и технологий.

№ п/п	Этап формирования (семестр)	Код формируемой компетенции (составляющей компетенции)		Критерии оценивания	Показатели оценивания (планируемые результаты обучения)		
					Пороговый уровень	Продвинутый уровень	Превосходный уровень
3.	3	ПК-17	ПК-17З	Теоретические навыки	Знать базовые технологии разработки объектов профессиональной деятельности для решения простых задач в различных областях.	Знать базовые технологии разработки объектов профессиональной деятельности для решения задач средней сложности в различных областях.	Знать базовые технологии разработки объектов профессиональной деятельности для решения сложных задач в различных областях.
4.	3	ПК-17	ПК-17У ПК-17В	Практические навыки	Уметь использовать базовые технологии разработки объектов профессиональной деятельности для решения простых задач в различных областях.	Уметь использовать базовые технологии разработки объектов профессиональной деятельности для решения задач средней сложности в различных областях.	Уметь использовать базовые технологии разработки объектов профессиональной деятельности для решения сложных задач в различных областях.
					Владеть базовыми технологиями разработки объектов профессиональной деятельности для решения простых задач в различных областях.	Владеть базовыми технологиями разработки объектов профессиональной деятельности для решения задач средней сложности в различных областях.	Владеть базовыми технологиями разработки объектов профессиональной деятельности для решения сложных задач в различных областях.

Формирование оценки при промежуточной аттестации по итогам освоения дисциплины зависит от уровня освоения компетенций, которые обучающийся должен освоить по данной дисциплине. Связь между итоговой оценкой и уровнем освоения компетенций (шкала оценивания) представлена в таблице 5.

Таблица 5

Описание шкалы оценивания

Описание оценки в требованиях к уровню и объему компетенций	Выражение в баллах	Словесное выражение
Освоен превосходный уровень усвоения компетенций	от 86 до 100	Зачтено (отлично)
Освоен продвинутый уровень усвоения компетенций	от 71 до 85	Зачтено (хорошо)
Освоен пороговый уровень усвоения компетенций	от 51 до 70	Зачтено (удовлетворительно)
Не освоен пороговый уровень усвоения компетенций	до 51	Не зачтено (не удовлетворительно)

5 Методические материалы, определяющие процедуру оценивания знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций

Формирование оценки по результатам текущего контроля успеваемости и промежуточной аттестации по итогам освоения дисциплины приведено в таблице 6.

Таблица 6

Формирование оценки по итогам освоения дисциплины (модуля) или практики

Наименование контрольного мероприятия	Рейтинговые показатели				
	I аттестация	II аттестация	III аттестация	по результатам текущего контроля	по итогам промежуточной аттестации
Раздел 1	**20**			**20**	
Тест текущего контроля по разделу	4			4	
Защита лабораторных работ	16			16	
Раздел 2		**20**		**20**	
Тест текущего контроля по разделу		4		4	
Защита лабораторных работ		16		16	
Промежуточная аттестация (экзамен):					**60**
– тест промежуточной аттестации по дисциплине					10
– в письменной форме по билетам					50

6 Контрольные задания или иные материалы, необходимые для оценки знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций в процессе освоения дисциплины

6.Тестовые задания

6.1. Тестовые задания. 1-ая часть

1.1. Алгоритм - это

- точное предписание по выполнению ограниченного числа определенных действий, последовательно преобразующих исходные данные в конечный результат;
- инструкция по выполнению некоторых действий преобразующих исходные данные;
- точное предписание по выполнению операторов программы;
- программа детально описывающая процедуру получения некоторого результата;
- формализованное описание заранее определенных действий, с помощью которых производится сложная обработка исходных данных;

1.2. Программа - это

- процесс преобразования исходных данных;
- набор операторов;
- алгоритм, предназначенный для выполнения на ЭВМ;
- специально подготовленный текст, понятный для ЭВМ;
- инструкция по выполнению некоторых действий преобразующих исходные данные;

1.3. Алгоритм обладает свойствами

- дискретность;
- непрерывность;
- бесконечность;
- конечность;
- массовость;
- одноразовость;
- результативность;
- неопределённость;
- запутанность;
- однозначность;
- продолжительность;

1.4. Базовыми алгоритмическими структурами являются

- структура последовательного выполнения действий;
- структура параллельного выполнения действий;
- структура выбора одного из многих действий;
- структура выбора одного из двух действий;
- структура многократного повторения действия с предварительной проверкой условия остановки;

- структура многократного повторения действия с последующей проверкой условия остановки;

1.5. Способами записи алгоритмов являются

- Словесный способ – на естественном языке;
- Графический (Блок-схемы);
- Псевдокод;
- Код на реальном языке программирования;
- Форма Бэкуса-Наура (БНФ);

1.6. Соответствие между базовыми типами данных языка С

	Целое число	Беззнаковое целое число	Вещественное число	Одиночный символ	Строка символов
float					
int					
long					
unsigned					
char					

1.7. Определение переменной Z для хранения данных о среднемесячной зарплате сотрудника (в рублях с точностью до копеек) имеет вид

- int Z;
- char Z;
- float Z;
- char Z[100];
- float *Z;

1.8. Описание данных для представления матрицы X, содержащей 10 строк по 15 вещественных чисел в строке может иметь вид:

- float X[15,10];
- float T[10,15];
- float X[10][15];
- float X[15][10];

1.9. Каким было значение целочисленной переменной x, если после выполнения следующих операторов оно стало равно 21?

/* x=? */ x++; x*=2; x--; x+=10; /* x=21 */

1.10. Дана целочисленная переменная N (99 < N <1000). Оператор, присваивающий целочисленной переменной X значение младшей десятичной цифры числа N имеет вид:

- X = N % 100 / 10;
- X = N / 100 % 10;
- X = N / 1000 * 10;
- X = N % 10;

1.11. Определение переменной для хранения данных о количестве учеников в каждом из 40 классов школы имеет вид

- int k[40][40];
- float k;
- float k[40];
- int k[40];

1.12. Определение переменной для хранения текста длиной до 80 символов имеет вид

- float t;
- float t[80];
- char t[80];
- char *t[80];
- float *t[80];

1.13. Значение переменной X после выполнения последовательности команд

int x,y; x=2; y=x*2; x++; y+=x; y=y*3; x=x+y;

равно ???

1.14. Значение переменной X после выполнения последовательности команд

int x,y,z; x=17; y=x/5; z=x%5; x=z+y;

равно ???

1.15. Операторы уменьшения на единицу величины Z

- Z = Z – 1;
- Z = - Z;
- Z = Z / 2;
- Z--;
- ++Z;

1.16. Оператор присваивания величине Y остатка от деления величины X на 3 имеет вид

- Y= X/3
- Y= X*3
- Y=X%3
- Y=%X3
- Y=X3%

1.17. Оператор, меняющий значение величины a на противоположное значение (например, 5 на -5), имеет вид

- a=a-1
- b=-a; a=-b;
- a=-a;
- a=1-a;

1.18. Соответствие между операторами языка С и их основным назначением

	Оператор цикла	Оператор возврата	Оператор ветвления
while			
int			
for			
do while			
if			
return			

1.19. Соответствие между функциями языка С и их основным назначением

	Ввод одного символа	Форматированный ввод данных	Вывод одного символа	Форматированный вывод данных	Ввод строки	Вывод строки
scanf()						
putchar()						
printf()						
getchar()						
gets()						

1.20. Условие проверки, является ли символ S латинской буквой, имеет вид

- if (s>'a' and s<'Z')
- if (s>'a' or s<'Z')
- if (s>='a' && s<='z' || s>='A' && s<='Z')
- if (s>'a' && s<'Z')
- if (s>'a' || s<'z)

1.21. Условие проверки, является ли значение целочисленной переменной x положительным и четным, имеет вид

- if (x>0 && x%5 == 0)
- if (x>0 || x%2)
- if (x<0 && x%2 == 0)
- if (x>0 && x%2 == 0)
- if (x>0 || x%2 == 0)

1.22. Условие в операторе ветвления if (x>4 && x<11 || x>=20)
будет истинно при значении переменной x равной

- 11
- 4
- 5
- 15

1.23. Оператор для присвоения переменной Z значения младшей цифры натурального числа X имеет вид

- X=Z%10;
- Z=X/10;
- Z=X%10;
- Z=X%1000;
- X=Z%2;

1.24. Значение целочисленной переменной a после выполнения фрагмента программы
int a,b; a = 2468; b= (a%1000)* 10; a=a/1000 + b;
равно ???

1.25. Результат работы фрагмента программы (программа на языке С)

```
int x, y;
scanf("%d%d",&x,&y);
printf("\n x = %d, y = %d, x / y = %d", x, y, x/y);
```

при вводе чисел 35 и 8 будет иметь вид ???

1.26. Результатом работы фрагмента программы (программа на языке С)

```
int x, c1, c2, c3, s;
scanf("%d",&x);
c1=x/100; c2= (x/10)%10; c3=x%10; s=c1+c2+c3;
printf("\n s= %d",s);
```

при вводе числа 354 является число ???

1.27. Значение переменной S после выполнения следующего фрагмента программы

```
int n,S, i;
n=5; S=0;
for(i=1; i<=n; i++)
    S=S+ i%3;
```

равно ???

1.28. Значение переменной S после выполнения следующего фрагмента программы (программа на языке С)

```
int n,S;
n=126; S=0;
while(n>0)
   { S=S+n%10;
    n=n/10;
    }
```

равно ???

1.29. Результатом работы фрагмента программы (программа на языке С)

```
int k=0; char s;
 while((s=getchar())!='.')
     if (s!=' ') k++;
```

является

- Количество символов в тексте, завершающемся точкой
- Количество пробелов в тексте, завершающемся точкой
- Количество символов, отличных от пробела, в тексте, завершающемся точкой
- Количество символов точка в заданном тексте

1.30. Объем памяти для данных определенных следующим образом
(при условии, что целая величина занимает 2 байта, вещественная 4 байта, указатели 2 байта памяти)

```
int x;   float y, *z;
```

равен ???

1.31. Количество повторений в заданном цикле оператора изменения значения величины S

```
int i; float s=0, x;
 for (i=0; i< 10; i++)
      { scanf("%f", &x);
        if (x >0 && x<11)   s=s+x;
      }
```

при вводе следующих величин

15, 10, 2, 20, 30, 12, 8, 1, 9, 10

равно ???

1.32. Значение переменной c после выполнения следующего фрагмента программы

```
int b=0,  c=0;
while (b!=10)
   { c=c+b;  b++; }
```

равно ???

1.33. Значение переменной k после выполнения следующего фрагмента программы

```
int k=0,  m;
for (m=1; m<=5; m++)
   if (m%3 != m%5)  k++;
```

равно ???

1.34. Структурное программирование - это

- программирование с использованием структур (записей)
- разбиение программы на подпрограммы
- разработка модульной структуры программы
- использование только структурных алгоритмов, составленных из базовых структур

1.35. Структурное программирование

- облегчает понимание программы
- уменьшает вероятность ошибок в программе
- уменьшает размер программы
- ускоряет разработку программы
- упрощает изменения программы

1.36. Хороший стиль программирования предусматривает

- комментарии в каждой строке программы
- использование осмысленных имен величин и функций
- максимальную экономию числа переменных
- ступенчатую запись вложенных операторов
- комментирование назначения каждой величины
- использование структурного программирования

1.37. Объем памяти для символьного массива: `char  str[] = “Привет!”;`
равен ???

1.38. Результатом выполнения фрагмента программы

```
char  s[80]=”Hello,  Kazan!!!”;
printf(“\n%s”,s);
```

является текст ???

1.39. Результатом выполнения фрагмента программы

```
char  s[80]=”KAI,  KGTU, KNITU!!!”;
puts(s);
```

является текст ???

1.40. Для массива характерно, что все элементы

- однотипны

- расположены по возрастанию
- связаны ссылками
- обозначены одинаковым именем и отличаются индексом (номером)
- одновременно присутствуют в оперативной памяти и одинаково доступны

1.41. Массивы используются для входных и/или выходных данных, если

- необходимо найти сумму и среднее значение элементов последовательности
- необходимо обрабатывать последовательность элементов неоднократно
- необходимо обрабатывать последовательность элементов не в том порядке, как они вводятся или выводятся
- необходимо найти максимальное и минимальное значения элементов последовательности

1.42. В результате работы фрагмента программы

```
int  x[10][10];  int i,  j; …..
for (i=0;   i<10;   i++)
    for ( j=0;  j<10;  j++)
        x[i][j] = i * 10 + j;
```

элемент массива x[1][2] будет равен ???

1.43. В результате работы фрагмента программы

```
int  x[10];  int i; …..
for (i=0;   i<10;   i++)
    x[i] =  i*i + 5;
```

элемент массива x[2] будет равен ???

1.44. В результате работы фрагмента программы

```
int   s,  x[100];  int  i; …..
for ( s=0, i=0;    i<100;    i++)
        { x[i] = s+i;  s = s + i; }
```

элементы массива x[5] будет равен ???

1.45. Значение переменной j после выполнения следующего фрагмента программы

```
int j;
. . .
for(j=0;j<10;j++)
m[j] = 0;
```

будет равно ???

1.46. Подпрограммы, используемые в языке С, могут быть в виде

- вычислимых функций
- функций, не обладающих значением
- функций, обладающих значением
- процедур, обладающих значением
- невычислимых функций

1.47. Переменная называется локальной, потому что

- ее можно использовать только в том блоке, где она объявлена
- она может иметь только целочисленные значения
- она не может изменяться

- она изменяется автоматически

1.48. В заголовке функции указываются

- только имена параметров
- только значения параметров
- типы и имена формальных параметров
- типы и имена только входных параметров
- типы и имена только выходных параметров
- типы и имена входных и выходных формальных параметров

1.49. Верное утверждение об операторе return

- в функции должен быть только один оператор return
- оператор return – оператор возврата, завершает выполнение функции
- оператор return обязательно последний оператор в теле функции
- оператор return означает аварийное завершение функции

1.50. Объявление функции, которому соответствует вызов функции f(10.5, &z);

- void f (int k, int x);
- int f (int k, int x);
- void f (float x, int *y);
- float f (float x, int y);
- int f (float k, int *m);

1.51. Объявлены переменные int k, m; float a, b, c; и функция int f (float x, float y, float *z);
Вызов функции может иметь вид

- k = f (a, b, c);
- k = f (a, b, &c);
- a = f (k, m, b);
- k = f (a, b, *c);
- m = f (a, b, &c);
- k = f (&a, &b, &c);

1.52. Для обмена значений двух переменных можно определить функцию

- void obmen (float *x, float *y) {float r; r = *x; *x = *y; *y = r;}
- void obmen (float *x, float *y) {int r; r = *x; *x = *y; *y = r;}
- void obmen (float x, float y) {float r; r = x; x = y; y = r;}
- void obmen (float *x, float *y) { *x = *y; *y = *x;}
- float obmen (float *x, float *y) {float r; r = *x; *x = *y; *y = r;}

1.53. Обязательно нужен оператор return в теле функции, объявление которой имеет вид

- f (int x, int y);
- f (float x, float y);
- float f (int n, float m[]);
- f (int n, float m[]);

1.54. Глобальная переменная

- объявлена как именованная константа
- может иметь только целочисленные значения
- не может изменяться
- изменяется автоматически
- может использоваться в любом блоке программы

1.55. Верное утверждение о необходимости прототипа функции

- прототип функции необходим, когда в программе несколько выходных параметров
- прототип функции необходим, когда в программе определение функции расположено после вызова этой функции
- прототип функции необходим, когда в программе нет определения функции
- прототип функции необходим, когда в программе несколько вызовов функции

1.56. Прототип функции - это

- имя функции
- значение функции
- заголовок функции, заканчивающийся точкой с запятой
- входные и выходные параметры функции

1.57. Фактические параметры задаются

- в прототипе функции
- при определении функции в заголовке
- при вызове функции
- в теле функции

1.58. Входные параметры функции можно передать

- по значению
- по умолчанию
- автоматически

1.59. Выходные параметры функции можно передать

- по значению
- по умолчанию
- по ссылке
- автоматически

6.2. Примеры экзаменационных заданий

6.2.1.Экзаменационные задания. 1-ая часть

2.1. Составить схему и трассировочную таблицу для данной программы.
Входной текст имеет вид: 28 70

```
#include <stdio.h>
main ()
{  int  x, y;
   scanf ("%d %d", &x, &y);
   while (x > 0 && y > 0)
     if (x > y)  x = x - y;
     else  y = y - x;
   if y = 0 then  printf ("%d", x);
   else  printf ("%d", y);
}
```

2.2. Составить схему и трассировочную таблицу для данной программы. Входной текст имеет вид: 24 80

```
#include <stdio.h>
main()
{ int a, b;
  scanf("%d %d", &a, &b);
   do
      if (b<=a)  a= a%b;
            else  b=b%a;
   while (a>0 && b > 0);
printf("%d",b);
}
```

2.3. Составить схему и трассировочную таблицу для данной программы. Входной текст имеет вид:
Allo.

```
#include <stdio.h>
main()
{ char x,y;
  y=getchar(); x=getchar();
 putchar(y);
 while(x!='.')
    {  if (x!=y) putchar(x);
       y=x;  x=getchar();
     }
}
```

6.2.2. Экзаменационные задания. 2-ая часть

3.1.Вычислить объем памяти для данных, определенных следующим образом:

```
int  r[50];

float  x;

char  t[] = "КГТУ";
```

3.2. Вычислить объем памяти для данных, определенных следующим образом:

```
float  z[10] [50];

int  x, *y;

char  s[80] = "КГТУ";
```

3.3. Вычислить объем памяти для данных, определенных следующим образом:

```
struct T

 {int  x[50];

   float  y;

   char f[25];

 } z, w[10],*adr;
```

6.2.3. Экзаменационные задания. 3-ая часть

4.1. Дано целое N>0 и последовательность из N действительных чисел. Составить программу опредения, сколько раз в этой последовательности *меняется знак*.

4.2. Дана последовательность действительных чисел, продолжающаяся до конца файла. Составить программу определения, *все ли* заданные числа *положительные*.

4.3. Дана последовательность из 500 целых чисел - количество очков каждого из 500 участников соревнований. Составить программу определения порядковых номеров участников, набравших максимальное количество очков. Составить схему и программу.

4.4. Дана последовательность действительных чисел, продолжающаяся до конца файла. Составить программу определения длины каждой последовательности положительных чисел.

4.5. Дан текст, завершающийся символом точка с запятой - ;. Составить программу для решения следующей задачи. Заменить в исходном тексте каждое сочетание символов != на сочетание символов ==. Подсчитать количество замен.

4.6. Входной текст, представляет собой исходный текст программы на языке С. Написать программу удаления из текста комментариев, ограниченных символами /* */.

4.7. Дано целое N>0 и N различных действительных чисел. Составить программу определения суммы элементов, расположенных между максимальным и минимальным числом.

4.8. Прямоугольная матрица X вещественных чисел размером n*m (0<n<=10, 0<m<=20) вводится по столбцам. Составить программу определения номера строкуис максимальной суммой элементов.

4.9. Дан текст произвольной длины, продолжающийся до конца файла. Составить программу определения, сколько различных литер (символов) входит в текст.

4.10. Дана последовательность символов, продолжающаяся до конца файла (при вводе с клавиатуры она заканчивается комбинацией клавиш Ctrl+Z). Составить программу выяснения, верно ли, что среди символов имеются все буквы слова "КГТУ".

4.11. Составить подпрограмму удаления заданного символа из заданной строки. Привести пример вызова подпрограммы.

6.2.4. Экзаменационные задания. 4-ая часть

5.1. Составить подпрограмму подсчета суммы и количества четных элементов в целочисленном одномерном массиве.

5.2. Составить подпрограмму определения среднего арифметического значения элементов кратных пяти в одномерном целочисленном массиве.

5.3. Составить подпрограмму увеличения положительных элементов в два раза и замены отрицательных элементов на число 0 в квадратной матрице размером 20x20.

Лист регистрации изменений

№ п/п	№ страницы внесения изменений	Дата внесения изменений	Содержание изменений	«Согласовано» Председатель УМК ИКТЗИ
1	2	3	4	6
1	1	01.02.2019	Изменение наименования учредителя университета. В соответствии с утверждением устава федерального государственного бюджетного образовательного учреждения высшего образования «Казанский национальный исследовательский университет им. А.Н. Туполева-КАИ» в новой редакции (Приказ № 1042 от 26.11.2018) наименование «Министерство образования и науки Российской Федерации» читать как «Министерство науки и высшего образования Российской Федерации»	
2				
3				
4				
5				