Государственное образовательное учреждение
«Приднестровский государственный университет им. Т.Г. Шевченко»

Физико-технический институт

Физико-математический факультет

Кафедра высшей и прикладной математики и информатики

УТВЕРЖДАЮ
Заведующий кафедрой высшей и
прикладной математики и информатики
_____ Коровай А.В.
(подпись) *(Ф.И.О.)*
протокол № 1 «30» 08 2024 г.

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

по дисциплине

Б1.О.15 «Программирование»

Направление

01.03.02 «Прикладная математика и информатика»

Профиль

Системное программирование и компьютерные технологии

Квалификация

Бакалавр

ГОД НАБОРА 2024

Разработчик: ст. преподаватель
_____ Калинкова Е.В.
(подпись) *(Ф.И.О.)*
«30» 08 2024 г.

г. Тирасполь 2024 г.

Паспорт фонда оценочных средств по учебной дисциплине

«Программирование»

1. В результате изучения дисциплины «Программирование» у обучающихся должны быть сформированы следующие компетенции:

Категория (группа) компетенций	Код и наименование	Код и наименование индикатора достижения компетенции
Универсальные компетенции и индикаторы их достижения		
Учебным планом не предусмотрены		
Общепрофессиональные компетенции и индикаторы их достижения		
Теоретические и практические основы профессиональной деятельности	ОПК-2 Способен использовать и адаптировать существующие математические методы и системы программирования для разработки и реализации алгоритмов решения прикладных задач	ИД ОПК-2.1. Обладает фундаментальными знаниями по существующим математическим методам и системам программирования для разработки и реализации алгоритмов решения прикладных задач.
		ИД ОПК-2.2. Умеет использовать аппарат существующих математических методов и систем программирования для разработки и реализации алгоритмов решения прикладных задач в профессиональной деятельности.
		ИД ОПК-2.3. Имеет навыки применения аппарата существующих математических методов и систем программирования для разработки и реализации алгоритмов при решении конкретных задач.
Информационно-коммуникационные технологии для профессиональной деятельности	ОПК-5 Способен разрабатывать алгоритмы и компьютерные программы, пригодные для практического применения	ИД ОПК-5.1. Знает методы алгоритмизации, языки и технологии программирования, пригодные для практического применения в области информационных технологий.
		ИД ОПК-5.2. Умеет применять методы алгоритмизации, языки и технологии программирования для решения задач профессиональной деятельности.
		ИД ОПК-5.3. Владеет навыками программирования, отладки и тестирования программных средств.
Обязательные профессиональные компетенции и индикаторы их достижения		
	ПК-1 Способен демонстрировать общенаучные базовые знания естественных наук, математики и информатики, понимание основных фактов, концепций, принципов теорий, связанных с прикладной математикой и информатикой	ИД ПК-1.1. Обладает базовыми знаниями, полученными в области математических и (или) естественных наук, программирования и информационных технологий.
		ИД К-1.2. Умеет находить, формулировать и решать стандартные задачи в собственной научно-исследовательской деятельности в области математических и (или) естественных наук, программирования и информационных технологий.
		ИД ПК-1.3. Имеет практический опыт научно-исследовательской деятельности в области математических и (или) естественных наук, программирования и информационных технологий.
	ПК-4 Способен демонстрировать знания современных языков программирования, операционных систем, офисных приложений, информационно-	ИД ПК-4.1. Знает основные языки программирования и основы работы с базами данных, операционные системы и оболочки, современные программные среды разработки информационных систем и технологий.
		ИД ПК-4.2. Умеет применять языки программирования, современные программные среды разработки информационных систем и

	телекоммуникационной сети "Интернет", способов и механизмов управления данными, принципов организации, состава и схемы работы операционных систем	технологий для автоматизации бизнес-процессов, решения прикладных задач различных классов, ведения баз данных и информационных хранилищ.
		ИД ПК-4.3. Владеет навыками программирования, отладки и тестирования прототипов программно-технических комплексов задач.
	ПК-5 Способен разрабатывать и применять алгоритмические и программные решения в области системного и прикладного программного обеспечения	ИД ПК-5.1. Знает разработку архитектуры, алгоритмических и программных решений системного и прикладного программного обеспечения.
		ИД ПК-5.2. Умеет использовать языки программирования, алгоритмы, библиотеки и пакеты программ, продукты системного и прикладного программного обеспечения.
		ИД ПК-5.3. Владеет навыками решения практических задач с применением языков программирования, алгоритмов, библиотек и пакетов программ.
Рекомендуемые профессиональные компетенции и индикаторы их достижения		
Учебным планом не предусмотрены		

2. Программа оценивания контролируемой компетенции:

1 семестр

Текущая аттестация	Контролируемые модули, разделы (темы) дисциплины и их наименование	Код контролируемой компетенции (или ее части)	Наименование оценочного средства
1	Раздел 1. Введение в язык программирования C#.	ОПК-2, ОПК-5, ПК-1, ПК-4, ПК-5	Комплект тестовых заданий № 1
2	Раздел 2. Управляющие операторы.	ОПК-2, ОПК-5, ПК-1, ПК-4, ПК-5	Комплект тестовых заданий № 2
3	Раздел 3. Массивы и строки.	ОПК-2, ОПК-5, ПК-1, ПК-4, ПК-5	Комплект тестовых заданий № 3
4	Раздел 4. Методы.	ОПК-2, ОПК-5, ПК-1, ПК-4, ПК-5	Комплект тестовых заданий № 4
5	Раздел 5. Классы. Инкапсуляция.	ОПК-2, ОПК-5, ПК-1, ПК-4, ПК-5	Комплект тестовых заданий № 5
6	Раздел 6. Организация иерархии классов.	ОПК-2, ОПК-5, ПК-1, ПК-4, ПК-5	Комплект тестовых заданий № 6
	Раздел 7. Интерфейсы и структурные типы.	ОПК-2, ОПК-5, ПК-1, ПК-4, ПК-5	Комплект тестовых заданий № 7
Промежуточная аттестация		Код контролируемой компетенции (или ее части)	Наименование оценочного средства
1		ОПК-2, ОПК-5, ПК-1, ПК-4, ПК-5	Список вопросов к экзамену

2 семестр

Текущая аттестация	Контролируемые модули, разделы (темы) дисциплины и их наименование	Код контролируемой компетенции (или ее части)	Наименование оценочного средства
1	Раздел 8. Обработка исключений. Раздел 9. Делегаты, лямбда-выражения, события.	ОПК-2, ОПК-5, ПК-1, ПК-4, ПК-5	Комплект тестовых заданий № 8
2	Раздел 10. Структуры данных.	ОПК-2, ОПК-5, ПК-1, ПК-4, ПК-5	Комплект тестовых заданий № 9
3	Раздел 11. Обобщенное программирование. Коллекции.	ОПК-2, ОПК-5, ПК-1, ПК-4, ПК-5	Комплект тестовых заданий № 10

	Linq.		
4	Раздел 12. Работа с файлами и файловой системой.	ОПК-2, ОПК-5, ПК-1, ПК-4, ПК-5	Комплект тестовых заданий № 11
5	Раздел 13. Визуальное программирование.	ОПК-2, ОПК-5, ПК-1, ПК-4, ПК-5	Комплект тестовых заданий № 12
Промежуточная аттестация		Код контролируемой компетенции (или ее части)	Наименование оценочного средства
1		ОПК-2, ОПК-5, ПК-1, ПК-4, ПК-5	Список вопросов к экзамену

Комплект тестовых заданий № 1
по дисциплине «Программирование»

Раздел 1. ***Введение в язык программирования C#***

1. Отличительными чертами языков программирования высокого уровня являются:

1) машинная независимость
2) форма записи программ, близкая к естественному языку
3) машинная зависимость
4) возможность прямого доступа к аппаратным ресурсам

2. Алфавит языка C# включает в себя:

1) символ подчеркивания
2) цифры от 0 до 9
3) лексемы
4) буквы латинского и национального алфавитов

3. Элементарные конструкции (лексемы) языка C# включают в себя:

1) идентификаторы
2) выражения
3) операторы
4) литералы (константы)
5) ключевые слова

4. Укажите допустимые идентификаторы в языке C#

1) Number One
2) x_1
3) result
4) _myName
5) 4_y
6) Dog1

5. Укажите правильные комментарии в языке C#

1) /* комментарий */
2) */ комментарий */
3) {комментарий}
4) ** комментарий **
5) // комментарий

6. Укажите ключевые слова для обозначения целых типов в языке C#

1) bool
2) byte
3) char
4) ushort
5) int
6) uint
7) long
8) float
9) double

7. Значение 256 может быть сохранено в переменной типа

1) bool
2) double
3) int
4) byte
5) short

8. Значение 5.9875e17 может быть сохранено в переменной типа

1) bool
2) double
3) int
4) long
5) short
6) необходимый тип отсутствует

9. Укажите бинарные операции

1) операции отношения

2) декремент
3) остаток от деления
4) условная операция
5) логическое отрицание

10. Укажите операции отношения (сравнения)

1) =
2) ==
3) !=
4) <
5) <<
6) >>
7) >=
8) +=

11. Что будет выведено на экран после выполнения фрагмента программы?

```
char a, b, c;
a = 'b';
b = 'c';
c = a;
Console.WriteLine("{0}{1}{2}", a, b, c);
```

1) abc
2) bcb
3) bca

12. Что будет выведено на экран после выполнения фрагмента программы?

```
Console.WriteLine(21 / 5 * 3);
```

1) 1
2) 1.4
3) 12
4) 12.6

Критерии оценки:

За каждое правильно выполненное задание назначается 1 балл. В случае задания с выбором нескольких правильных ответов 1 балл дается за полностью выполненное задание, 0 баллов – за хотя бы один неверный ответ. Затем первичные баллы (максимум – 12 баллов) пропорционально переводятся в тестовые баллы согласно технологической карте дисциплины.

Комплект тестовых заданий № 2
по дисциплине «Программирование»

Раздел 2. ***Управляющие операторы***

1. Чему равно значение выражения (`!a && (b||c)`), где a, b и c – величины типа bool, имеющие значения `true`, `true` и `false` соответственно?

1) 0
2) 1
3) false
4) true

2. Работа какого оператора дает верный ответ, при условии, что f1, f2 – переменные логического типа?

1)
```
if (!f1 && f2 || f1 && !f2)
    Console.WriteLine("только одна из f1,f2  имеет значение истина");
```
2)
```
if (!f1 && f2 || f1 && !f2)
    Console.WriteLine("хотя бы одна из f1,f2 имеет значение истина");
```
3)
```
if (!f1 && f2 || f1 && !f2)
    Console.WriteLine("переменные f1,f2 имеют одинаковое значение");
```

3. Работа каких операторов дает верный ответ?

1)
```
if (x/2==0) Console.WriteLine("x четное");
else Console.WriteLine("x нечетное");
```
2)
```
if (x%2==0) Console.WriteLine("x четное");
else Console.WriteLine("x нечетное");
```
3)
```
if (x%2!=0) Console.WriteLine("x нечетное");
else Console.WriteLine("x четное");
```
4)
```
if (x%2!=0) Console.WriteLine("x четное");
else Console.WriteLine("x нечетное");
```

4. Определите значение переменной *c* после выполнения следующего фрагмента программы:

```
int a = 100, b = 30;
a = a – b * 3;
if (a > b)  c = a – b;
else c = b – a;
```

1) -80
2) -70
3) 70
4) 20

5. Результат работы фрагмента программы

```
for (int i=1; i<=3; i++)
{
    Console.Write("i");
}
```

1) 123
2) iii
3) фрагмент кода содержит ошибку
4) тело цикла не выполнится ни разу

6. Чему будет равно значение переменной m после выполнения фрагмента программы?

```
int k = 3, m = 0;
while (k > 0) k--; m++;
```

1) 1
2) 2
3) 3
4) 4

7. Результат работы фрагмента программы

```
int x = 10;
while (x < 0)
{
    Console.Write("{0} ",x);
```

```
    x = x - 3;
}
```

1) 10 7 4
2) 10 7 4 1
3) тело цикла не выполнится ни разу
4) фрагмент кода содержит ошибку

8. Результат работы фрагмента программы

```
int x = 10;
do
{
    Console.Write("{0} ", x);
    x = x - 3;
}
while (x >= 1);
```

1) 10 7 4 1
2) 10 7 4
3) 10
4) тело цикла не выполнится ни разу

9. Что будет выведено на экран в результате выполнения фрагмента программы?

```
for (int i = 0; i < 3; i++)
{
   switch (i)
   {
      case 0 : Console.Write("ZERO "); break;
      case 1 : Console.Write("ONE "); break;
      default: Console.Write("DEF "); return;
   }
}
```

1) ZERO ONE
2) ZERO ONE DEF
3) ZERO DEF DEF
4) ZERO ONE ONE DEF
5) ZERO ONE DEF DEF

10. Что будет выведено в результате работы следующего фрагмента программы?

```
int i = 17;
while (i != 1)
{
    Console.Write("{0} ", i);
    i = 3 * i + 1;
    while (i % 2 == 0)
      i /= 2;
}
```

1) 17 15 13 11 9 7 5 3 1
2) 17 13 5 4 1
3) 17 13 5 4
4) 17 13 5

11. Что будет выведено в результате работы следующего фрагмента программы?

```
int y;
for (int x = 1; x <= 10; x++)
{
    if (x % 2 != 0) continue;
    y = 2 * x;
    Console.Write(y + " ");
}
```

1) 2 4 6 8 10 12 14 16 18 20
2) 2 6 10 14 18
3) 4 8 12 16 20
4) ничего

12. Что будет выведено в результате работы следующего фрагмента программы?

```
int y;
for (int x = 1; x <= 10; x++)
```

```
{
   if (x % 5 == 0) break;
   y = x + 2;
   Console.Write(y + " ");
}
```

1) 3 4 5 6 8 9 10 11
2) 3 4 5 6
3) 3 4 5 6 7 8 9 10 11 12
4) ничего

Критерии оценки:

За каждое правильно выполненное задание назначается 1 балл. В случае задания с выбором нескольких правильных ответов 1 балл дается за полностью выполненное задание, 0 баллов – за хотя бы один неверный ответ. Затем первичные баллы (максимум – 12 баллов) пропорционально переводятся в тестовые баллы согласно технологической карте дисциплины.

Комплект тестовых заданий № 3
по дисциплине «Программирование»

Раздел 3. ***Массивы и строки***

1. Массив объявлен следующим образом

```
int[] myArray = new int[10];
```

Какой из следующих фрагментов программы будет правильно выводить элементы данного массива:

1)
```
for (i = 1; i <= 10; i++)
Console.WriteLine(myArray[i]);
```
2)
```
for (i = 0; i <= 10; i++)
Console.WriteLine(myArray[i]);
```
3)
```
for (i = 0; i < 10; i++)
Console.WriteLine(myArray[i]);
```
4)
```
for (i = 1; i < 10; i++)
Console.WriteLine(myArray[i]);
```

2. Массив объявлен следующим образом:

```
int[ , , ] arr = new int[5,2,3];
```

Чему равно значение `arr.Length`?

1) 3
2) 5
3) 10
4) 30
5) Не определено

3. Массив объявлен следующим образом:

```
int[,] MyArray = new int[4, 7];
```

Какой метод возвращает количество столбцов данного массива?

1) `MyArray.GetLength(0)`
2) `MyArray.GetLength(1)`
3) `MyArray.Rank(0)`
4) `MyArray.Length[0]`
5) `MyArray.Length[1]`

4. Массив инициализирован следующим образом:

```
int[,] a = {{4, 5}, {1, 3}};
```

Элемент `a[1,1]` имеет значение:

1) 4
2) 5
3) 1
4) 3

5. Значения массива задаются с помощью следующего фрагмента программы:

```
int[] a = new int[6];
a[0] = 3;
for (int i = 1; i < a.Length; i++)
    a[i] = 4 - a[i - 1];
```

Чему равен последний элемент массива?

6. Чему равно значение переменной s после выполнения фрагмента программы:

```
int[,] a = { { 1, 2, 3 }, { 4, 5, 6 }, { 7, 8, 9 } };
int s = 0;
foreach (int i in a)
    s += i;
```

7. Что будет выведено на экран в результате работы следующего фрагмента программы?

```
int[] a = {3, 8, 0, -6, 0, -1, -9, 3};
int i, k;
```

```
k = a[0];
for (i = 1; i < 8; i++)
     if (a[i] > k) k = a[i];
Console.WriteLine(k);
```

8. Что будет выведено на экран в результате работы следующего фрагмента программы?

```
int[] a = {3, 8, 0, -6, 0, -1, -9, 3};
int i, k;
k = Math.Abs(a[7]);
for (i = 0; i< 8; i++)
    if (Math.Abs(a[i]) > k) k = Math.Abs(a[i]);
Console.WriteLine(k);
```

9. Какая величина вычисляется во фрагменте программы?

```
int[] a = {3, 8, 0, -6, 0, -1, -9, 3};
int k = 0;
for (int i = 7; i >= 0; i--)
    if (a[i] < 0) k += i;
Console.WriteLine(k);
```

1) сумма отрицательных элементов массива
2) количество номеров отрицательных элементов массива
3) сумма номеров отрицательных элементов массива
4) количество отрицательных элементов массива

10. Что будет выведено на экран после выполнения следующего программного фрагмента

```
int [ , ] array= new int[4,4];
int i, j;
for (i = 0; i < 4; i++)
    for (j = 0; j < 4; j++)
        array[i,j] = j;
for (i = 0; i < 4; i++)
{
    for (j = 0; j < 4; j++)
        Console.Write(array[i, j]);
}
```

1) 0123012301230123
2) 1234123412341234
3) 3210321032103210
4) 4321432143214321
5) 0000111122223333

11. В результате выполнения фрагмента программы:

```
char [] a={'м', 'и', 'р', 'а', 'ж'};
string line=new string (a, 1, 3);
Console.WriteLine(line);
```

на экран будет выведено:

1) мир
2) ира
3) ничего, т.к. фрагмент содержит ошибку

12. Что будет выведено на экран в результате выполнения фрагмента программы?

```
char[] a = {'a', 'b', 'c', 'r', 'c', 'a', 'a', 'b'};
int k = 0;
for (int i = 0; i < 8; i++)
   if (a[i] > 'c') k++;
Console.Write(k);
```

1) 0
2) 1
3) 2
4) 3
5) 5

13. Что будет выведено на экран в результате выполнения фрагмента программы?

```
string str = "кол около колокола";
str.Replace("о","");
Console.WriteLine(str);
```

1) кл кл клкла
2) кл около колокола
3) кол около колокола

14. В результате какой последовательности операторов на экран будет выведена строка «ученье – свет»

1)
```
StringBuilder s = new StringBuilder();
s.Append("ученье тьма" ) ;
s.Remove(7,4);
s.Append(" свет" ) ;
s.Insert(7,'-');
Console.WriteLine(s);
```

2)
```
String s = new String();
s.Append("ученье тьма" ) ;
s.Remove(7,4);
s.Append(" свет" ) ;
s.Insert(7,'-');
Console.WriteLine(s);
```

3)
```
StringBuilder s = new StringBuilder();
s.Append("ученье тьма" ) ;
s.Remove(8,4);
s.Append(" свет" ) ;
s.Insert(8,'-');
Console.WriteLine(s);
```

4)
```
StringBuilder s = new StringBuilder();
s.Append("ученье тьма" ) ;
s.Remove(7,4);
s.Insert(7,'-');
s.Append("свет" ) ;
Console.WriteLine(s);
```

Критерии оценки:

За каждое правильно выполненное задание назначается 1 балл. Затем первичные баллы пропорционально переводятся в тестовые баллы согласно технологической карте дисциплины.

Комплект тестовых заданий № 4
по дисциплине «Программирование»

Раздел 4. ***Методы***

1. Что будет выведено на экран в результате выполнения фрагмента программы?

```
class Program
{
    static int F(int a, int b)
    {
        return 2 * a + b;
    }
    static void Main()
    {
        int a = 1, b = 5;
        Console.WriteLine(F(b, a));
    }
}
```

1) 7
2) 11
3) 12

2. Что будет выведено на экран в результате выполнения фрагмента программы?

```
class Program
{
    static void F(int a)
    {
        a++;
    }
    static void Main()
    {
        int a=5;
        F(a);
        Console.WriteLine(a);
    }
}
```

1) 0
2) 5
3) 6

3. Спецификатор `ref` используется для передачи параметра:

1) в качестве выходного параметра
2) по значению
3) по ссылке

4. Если мы объявили метод `void Func(int x, out int y)`, какой из вызовов правильный?

1) Func(a, out b)
2) Func(a, b)
3) Оба

5. Что будет выведено на экран в результате выполнения фрагмента программы?

```
class Program
{
    static int F(int x, int y)
    {
        if (x < y) return x;
        else return y;
    }
    static void Main()
    {
        int a = 7, b = 5, c = 9;
        int y = F(2 * a, b - c);
        int z = F(F(a, b), c);
        Console.WriteLine("{0} {1}", y, z);
    }
}
```

1) 14 9
2) -4 9
3) -4 5
4) 14 6

6. Что будет выведено на экран в результате выполнения фрагмента программы?

```
class Program
{
    static void F(int x, ref int y)
    {
        x += 5; y += 5;
        Console.WriteLine("{0} {1}", x, y);
    }
    static void Main()
    {
        int a = 10, b = 10;
        F(a, ref b);
        Console.WriteLine("{0} {1}", a, b);
    }
}
```

1) 10 10
2) 10 15
3) 10 15
   10 10
4) 10 15
   10 15
5) 15 15
   10 15

7. Что будет выведено на экран в результате выполнения фрагмента программы?

```
class Program
{
    static void F(ref int x, ref int y)
    {
        int t;
        t = x; x = y; y = t;
    }
    static void Main()
    {
        int a = 5, b = 3, c = 9;
        F(ref a, ref b);
        F(ref b, ref c);
        Console.WriteLine("{0} {1} {2}", a, b, c);
    }
}
```

1) 5 3 9
2) 3 5 9
3) 5 9 3
4) 3 9 5

8. Дан рекурсивный метод:

```
static int F(int n)
{
    if (n > 10) return n + F(n - 10);
    else return 0;
}
```

Чему равно `F(42)`?

Критерии оценки:

За каждое правильно выполненное задание назначается 1 балл. Затем первичные баллы пропорционально переводятся в тестовые баллы согласно технологической карте дисциплины.

Комплект тестовых заданий № 5
по дисциплине «Программирование»

Раздел 5. ***Классы. Инкапсуляция.***

9. В объектно-ориентированном программировании правильным является утверждение:

1) объект – это то же самое, что и класс
2) класс и объект – не связанные между собой понятия
3) объект – это экземпляр класса
4) класс – это экземпляр объекта

10. Сокрытие информации и комбинирование данных и методов внутри объекта определяет понятие …

1) иерархии
2) инкапсуляции
3) полиморфизма
4) наследования

11. Среди перечисленных конструкций укажите объявление метода

1) `string GetName() {return "Name";}`
2) `string Name { get {return "Name";} }`
3) `string this [int i] { get {return "Name";} }`
4) `string Name;`

12. Среди перечисленных конструкций укажите объявление поля

1) `string GetName() {return "Name";}`
2) `string Name { get {return "Name";} }`
3) `string this [int i] { get {return "Name";} }`
4) `string Name;`

13. Что будет выведено на экран в результате выполнения программы?

```
using System;
class Person
{
    int age = 26;
    string name = "Tom";
    public Person(int age, string name)
    {
        this.age = age;
        this.name = name;
    }
}
class Program
{
    static void Main(string[] args)
    {
        Person person = new Person(19, "Bob");
        Console.WriteLine(person.name);
    }
}
```

1) `Bob`
2) `Tom`
3) программа не скомпилируется

14. Для инкапсуляции вы хотите определить автосвойство так, чтобы невозможно было вызвать сеттер за пределами класса, но геттер оставить доступным. В каком из заданных вариантов описано нужное свойство?

1) `public int Value { get; }`
2) `public int Value { get; private set; }`
3) `public int Value { get; set; }`
4) `private int Value { public get; set; }`
5) `int Value { public get; private set; }`

15. Укажите, какие свойства определены верно.

```
public class SomeClass
{
    private int a;
    private int c;

    public int A
    {
        get { return a; }
        set { a = value; }
    }

    public int B
    {
        get { return 0; }
    }

    public int C
    {
        set { c = value; }
    }

    protected int D { get; set; }

    protected int E { public get; set; }
}
```

1) A
2) B
3) C
4) D
5) E

16. Для одного класса может быть определено несколько конструкторов при условии что:

1) все они имеют различные имена
2) все они имеют различные списки параметров
3) по крайней мере один из них является конструктором по умолчанию
4) по крайней мере один из них инициализирует все поля

17. Перегрузка методов означает:

1) они имеют разные имена
2) они имеют одинаковые имена, но относятся к разным классам иерархии
3) они имеют одинаковые имена и одинаковые списки параметров
4) они имеют одинаковые имена и различные списки параметров

18. Что будет выведено на экран в результате выполнения программы?

```
using System;

class Person
{
    public string Name, Family;
    public byte Age;

    public Person(string name, string family)
    {
        Name = name;
        Family = family;
        Console.Write(1);
    }

    public Person(string name, string family, byte age):
        this(name, family)
    {
        Age = age;
        Console.Write(2);
    }
```

```
}

class Program
{
    static void Main(string[] args)
    {
        Person user = new Person("Александр", "Ерохин", 26);
        Console.ReadLine();
    }
}
```

1) ничего
2) 1
3) 2
4) 12
5) 21

Критерии оценки:

За каждое правильно выполненное задание назначается 1 балл. В случае задания с выбором нескольких правильных ответов 1 балл дается за полностью выполненное задание, 0 баллов – за хотя бы один неверный ответ. Затем первичные баллы (максимум – 10 баллов) пропорционально переводятся в тестовые баллы согласно технологической карте дисциплины.

Комплект тестовых заданий № 6
по дисциплине «Программирование»

Раздел 6. ***Организация иерархии классов.***

1. Для объектно-ориентированной технологии программирования наследование – это

1) способность объекта сохранять свойства и методы класса-родителя
2) сокрытие информации и комбинирование данных и методов внутри объекта
3) возможность задания в иерархии объектов различных действий в методе с одним именем
4) заключение в отдельный модуль процедур работы с объектом

2. Полиморфизм характеризуется

1) сокрытием информации и комбинированием данных и методов внутри объекта
2) способностью объекта наследовать свойства и методы класса-родителя
3) посылкой сообщений объектам
4) возможностью задания в иерархии объектов различных действий в методе с одним именем

3. Определены четыре класса

```
class ClassA
{
  public string fieldA;
}
class ClassB : ClassA
{
  public string fieldB;
}
class ClassC : ClassA
{
  public string fieldC;
}
class ClassD : ClassC
{
  public string fieldD;
}
```

Какие поля будет содержать каждый экземпляр класса `ClassC`?

1) `fieldA`
2) `fieldB`
3) `fieldC`
4) `fieldD`

4. Определены четыре класса

```
class ClassA
{
  public string fieldA;
}
class ClassB : ClassA
{
  public string fieldB;
}
class ClassC : ClassA
{
  public string fieldC;
}
class ClassD : ClassC
{
  public string fieldD;
}
```

Какие поля будет содержать каждый экземпляр класса `ClassD`?

1) `fieldA`
2) `fieldB`
3) `fieldC`
4) `fieldD`

5. Класс `classA` имеет два поля (типа `int`) `field1` и `field2` и конструктор, который инициализирует эти поля. Класс `classB` унаследован от `classA`. Он имеет собственное поле `field3`. Как надо определить конструктор класса `classB`, чтобы он инициализировал все три поля:

1)
```
public classB (int a, int b, int c)
{
     classA(a, b);
     field3 = c;
}
```
2)
```
public classB (int a, int b, int c)
{
     base(a, b);
     field3 = c;
}
```
3)
```
public classB (int a, int b, int c): base(a, b)
{
    field3 = c;
}
```
4)
```
public classB (int a, int b, int c)
{
     super(a, b);
     field3 = c;
}
```

6. Если метод объявлен как виртуальный, тогда любой производный класс:

1) может представить переопределенный вариант этого метода с точно же такими параметрами
2) должен представить переопределенный вариант этого метода с точно же такими параметрами
3) может представить переопределенный вариант этого метода с теми же или другими параметрами
4) должен представить переопределенный вариант этого метода с теми же или другими параметрами

7. Определены два класса

```
class classA
{
    int field1;
    int field2;
}
class classB:classA
{
    public int field3;
    public classB(int a, int b)
    {
        field1 = a;
        field2 = b;
        field3 = field1 + field2;
    }

    public void Show()
    {
        Console.WriteLine(field3);
    }
}
```

Что будет на экране после выполнения следующего фрагмента:

```
classB ob1 = new classB(2,3);
ob1.Show();
```

1) 5
2) ошибка на этапе компиляции из-за того, что не создан экземпляр класса `classA`
3) ошибка на этапе компиляции из-за того, что закрытые члены базового класса `field1` и `field2` не доступны в производном классе

8. Какие из определений для абстрактных классов являются корректными?

1) абстрактные классы нельзя наследовать
2) нельзя создавать экземпляр абстрактного класса
3) в абстрактных классах нельзя определять поля
4) абстрактные члены классов не должны иметь модификатор private

5) если класс имеет хотя бы один абстрактный метод (или абстрактные свойство, индексатор, событие), то он должен быть определен как абстрактный

9. В результате выполнения следующей программы:

```
using System;

abstract class Base
{
    protected string name = "Base";
    public abstract void Display();
    public abstract void Do();
}
class Derived : Base
{
    public override void Display()
    {
        Console.WriteLine(name);
    }
}

class Program
{
    static void Main(string[] args)
    {
        Derived b = new Derived();
        b.Display();
    }
}
```

1) появится текст «Base»
2) произойдет ошибка компиляции из-за того, поле name имеет модификатор доступа protected
3) произойдет ошибка компиляции из-за того, что производный неабстрактный класс Derived должен реализовать все абстрактные методы класса Base
4) произойдет ошибка компиляции из-за того, что производный неабстрактный класс Derived не содержит конструктор

Критерии оценки:

За каждое правильно выполненное задание назначается 1 балл. В случае задания с выбором нескольких правильных ответов 1 балл дается за полностью выполненное задание, 0 баллов – за хотя бы один неверный ответ. Затем первичные баллы (максимум – 9 баллов) пропорционально переводятся в тестовые баллы согласно технологической карте дисциплины.

Комплект тестовых заданий № 7
по дисциплине «Программирование»

Раздел 7. ***Интерфейсы и структурные типы.***

1. Каким модификатором неявно снабжаются все элементы интерфейса?

1) public
2) abstract
3) private
4) static
5) protected

2. Выберите правильное объявление интерфейса:

1)
```
public interface IPoint
{
    public void Show();
    public double Distance();
}
```
2)
```
public interface IPoint
{
    void Show();
    double Distance();
}
```
3)
```
public interface IPoint
{
    double x, y;
    void Show();
    double Distance();
}
```
4)
```
public interface IPoint
{
    public double x, y;
    public void Show();
    public double Distance();
}
```

3. Выберите верные утверждения

1) в интерфейсе можно объявлять поля
2) в интерфейсе можно указывать свойства, индексаторы и события
3) в интерфейсе можно реализовывать методы
4) можно создать экземпляр интерфейса
5) интерфейсы по умолчанию имеют уровень доступа internal
6) в интерфейсе могут быть конструкторы

4. Что будет выведено на консоль в результате выполнения следующего кода:

```
using System;

class Person
{
    public string name;
}

class Program

{
    static void Main(string[] args)
    {
        Person p1 = new Person { name = "Александр"};
        Person p2 = new Person { name = "Александр" };
        Person p3 = p1;
        Console.WriteLine(p1.Equals(p2));
        Console.WriteLine(p1.Equals(p3));
        Console.ReadLine();
    }
}
```

1)
```
True
True
```
2)
```
True
False
```
3)
```
False
True
```
4)
```
False
False
```

5. Что будет выведено на консоль в результате выполнения следующего кода:

```
using System;

struct Person
{
    public string name;
}

class Program
{
    static void Main(string[] args)
    {
        Person p1 = new Person { name = "Александр"};
        Person p2 = new Person { name = "Александр" };
        Person p3 = p1;
        Console.WriteLine(p1.Equals(p2));
        Console.WriteLine(p1.Equals(p3));
        Console.ReadLine();
    }
}
```

1)
```
True
True
```
2)
```
True
False
```
3)
```
False
True
```
4)
```
False
False
```

6. Выберите правильные утверждения:

1) объект структуры нельзя создавать с помощью оператора new
2) структуры не могут наследовать другие структуры
3) структуры не могут содержать методы
4) структуры – это типы значений
5) для структуры можно определить конструктор без параметров
6) для структуры нельзя определить несколько конструкторов

7. Какие из следующих высказываний правильные?

1) объект структуры можно создавать без оператора new, но тогда он будет неинициализирован
2) объект структуры можно создавать только с помощью оператора new
3) если объект структуры создается с помощью оператора new, то обязательно вызывается конструктор по умолчанию
4) если объект структуры создается с помощью оператора new, то поля структуры будут инициализированы либо конструктором по умолчанию, либо конструктором, определенным пользователем

Критерии оценки:

За каждое правильно выполненное задание назначается 1 балл. В случае задания с выбором нескольких правильных ответов 1 балл дается за полностью выполненное задание, 0 баллов – за хотя бы один неверный ответ. Затем первичные баллы (максимум – 7 баллов) пропорционально переводятся в тестовые баллы согласно технологической карте дисциплины.

Комплект тестовых заданий № 8
по дисциплине «Программирование»

Раздел 8, 9. ***Обработка исключений. Делегаты, лямбда-выражения, события.***

1. В результате выполнения следующего фрагмента программы

```
int x = 10, y = 0;
try
{
     Console.WriteLine("Частное = " + x / y);
}
catch (DivideByZeroException)
{
     Console.WriteLine("Деление на ноль");
}
catch (Exception)
{
     Console.WriteLine("Сгенерировано исключение");
}
```

1) появится сообщение «Деление на ноль»
2) появится сообщение «Сгенерировано исключение»
3) произойдет ошибка компиляции
4) программа выполнится, но на консоль ничего выведено не будет

2. Что будет выведено на экран в результате выполнения следующего фрагмента программы?

```
int x = 7, y = 0;
try
{
    int z= x / y;
    Console.Write("1");
}
catch (Exception)
{
    Console.Write("2");
}
finally
{
    Console.Write("3");
}
```

1) 12
2) 13
3) 2
4) 23
5) 123

3. Какое значение примет s в результате выполнения следующего программного фрагмента, если мы будем вводить с клавиатуры -4, 5, -6, 10, -5:

```
int x, s =0 ;
for (int i = 1; i <= 5; i++)
{
   try
   {
       x = Convert.ToInt32(Console.ReadLine());
       if (x > 0) throw new Exception();
       s += x;
    }
    catch (Exception)
    {
        Console.WriteLine("введено недопустимое значение");
    }
 }
 Console.WriteLine("s=" + s);
```

1) -4
2) -10
3) -15

4) 0
5) -5

4. Что будет выведено на экран после выполнения следующего программного фрагмента

```
try
{
    Console.WriteLine("Введите два числа");
    int x = Convert.ToInt32(Console.ReadLine());
    int y = Convert.ToInt32(Console.ReadLine());
    Console.WriteLine(x / y);
}
catch (FormatException)
{
    Console.WriteLine("Ошибка. Вы ввели не целое число.");
}
catch (OverflowException)
{
    Console.WriteLine("Ошибка. Вы ввели слишком длинное число.");
}
```

если с клавиатуры будут введены числа

12345678987654321
12345678987654321

1) Ошибка. Вы ввели не целое число.
2) Ошибка. Вы ввели слишком длинное число.
3) 1
4) случится аварийное прерывание

5. Какой оператор позволяет сгенерировать исключение вручную?

1) throw
2) finally
3) exception
4) try
5) catch

6. Есть следующий делегат:

```
delegate int Operation (int val);
```

Какой из следующих методов соответствует данному делегату:

1)
```
static void Method1(int x)
{
    Console.WriteLine(x * x);
}
```
2)
```
static int Method2(ref int x)
{
    return x * x;
}
```
3)
```
static int Method3(int x, int y)
{
    return x * y;
}
```
4)
```
static int Method4(int x, int y = 7)
{
    return x * y;
}
```
5)
```
static int Method5(int x)
{
    return x * x;
}
```
6)
```
static int Method6(out int x)
{
    x = 7;
    return x * x;
}
```

7. Какой будет консольный вывод при выполнении следующей программы:

```
using System;

class Program
{
    delegate void Message();

    static void Main(string[] args)
    {
        Message mes1 = Hello;
        mes1 += HowAreYou;
        mes1 += Hello;
        mes1 += Hello;
        mes1 -= Hello;
```

```
        mes1();
        Console.ReadLine();
    }

    private static void Hello() { Console.WriteLine("Hello"); }
    private static void HowAreYou() { Console.WriteLine("How are you?"); }
}
```

1) How are you?
2) Hello
   How are you?
3) Hello
   How are you?
   Hello
4) How are you?
   Hello

8. Что будет выведено на консоль в результате выполнения следующей программы:

```
using System;

class Program
{
    delegate int Operation(int x, int y);

    static void Main(string[] args)
    {
        Operation op = Subtract;
        Console.WriteLine(op(9, 5));
        op = Add;
        Console.WriteLine(op(4, 7));
        Console.ReadLine();
    }
    private static int Subtract(int x, int y) { return x - y; }
    private static int Add(int x, int y) { return x + y; }
}
```

1) 4
2) 11
3) 4
   11

9. Что будет выведено на консоль в результате выполнения следующей программы:

```
using System;

class Program
{
    delegate int Operation(int x, int y);

    static void Main(string[] args)
    {
        Operation del = Add;
        del += Multiply;
        int result = del(6, 5);
        Console.WriteLine(result);
        Console.ReadLine();
    }
    private static int Add(int x, int y) { return x + y; }
    private static int Multiply(int x, int y) { return x * y; }
}
```

1) 11
2) 30
3) 11
   30

10. Что будет выведено на консоль в результате выполнения следующей программы:

```
using System;
```

```
class Program
{
    static void Main(string[] args)
    {
        int[] array = { 1, 2, 3, 4, 5, 6, 7, 8, 9 };
        int result = F(array, x => x % 2 != 0);
        Console.WriteLine(result);
        Console.ReadLine();
    }

    private static int F(int[] numbers, Predicate<int> func)
    {
        int result = 0;
        foreach (int x in numbers)
        {
            if (func(x))
                result += x;
        }
        return result;
    }
}
```

11. Что будет выведено на консоль в результате выполнения следующей программы:

```
using System;

class Program
{
    static void Main(string[] args)
    {
        Console.WriteLine(F(4, x => x * x));
        Console.ReadLine();
    }

    static double F(int n, Func<double, double> f)
    {
        double s = 0;
        for (int k = 1; k <= n; k++)
            s += f(k);
        return s;
    }
}
```

Критерии оценки:

За каждое правильно выполненное задание назначается 1 балл. Затем первичные баллы пропорционально переводятся в тестовые баллы согласно технологической карте дисциплины.

Государственное образовательное учреждение
«Приднестровский государственный университет им. Т.Г. Шевченко»

Физико-математический факультет
Кафедра прикладной математики и информатики

Комплект тестовых заданий № 9
по дисциплине «Программирование»

Раздел 10. ***Структуры данных.***

1. Что характерно для динамической реализации структур данных?

1) использование адресных переменных (указателей/ссылок) для связывания элементов структуры
2) возможность выделения памяти для элементов структуры во время выполнения программы
3) использование массивов как основы реализации
4) распределение памяти под элементы структуры во время компиляции программы

2. Какие утверждения относительно динамической реализации списков являются правильными?

1) каждому элементу списка во время выполнения программы выделяется своя область памяти
2) каждый элемент списка имеет специальное поле с адресом следующего элемента
3) логический порядок следования элементов в списке может не совпадать с физическим размещением элементов в памяти
4) максимальное число элементов в списке должно быть известно заранее

3. Структура данных, работа с элементами которой организована по принципу FIFO (первый пришел – первый ушел), это:

1) стек
2) дек
3) очередь
4) список

4. В чём особенность стека?

1) открыт с обеих сторон на вставку и удаление
2) открыт с одной стороны на вставку и удаление
3) доступен любой элемент

5. В чём особенности очереди?

1) открыта с обеих сторон
2) открыта с одной стороны на вставку и удаление
3) доступен любой элемент

6. Линейный последовательный список, в котором включение и исключение элементов возможно с обоих концов, называется

1) стеком
2) очередью
3) деком
4) кольцевой очередью

7. Элемент дерева, который не ссылается на другие, называется

1) корнем
2) листом
3) узлом
4) промежуточным

8. Элемент дерева, на который не ссылаются другие, называется

1) корнем
2) листом
3) узлом
4) промежуточным

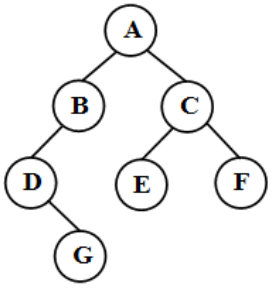
9. Высотой дерева называется

1) максимальное количество узлов
2) максимальное количество связей
3) максимальное количество листьев
4) максимальная длина пути от корня до листа

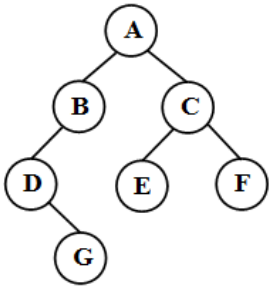
10. Степенью дерева называется

1) максимальная степень узла, входящего в дерево
2) максимальное количество уровней его узлов
3) максимальное количество узлов
4) максимальное количество связей
5) максимальное количество листьев

11. Для дерева, изображенного на рисунке, укажите высоту вершины B



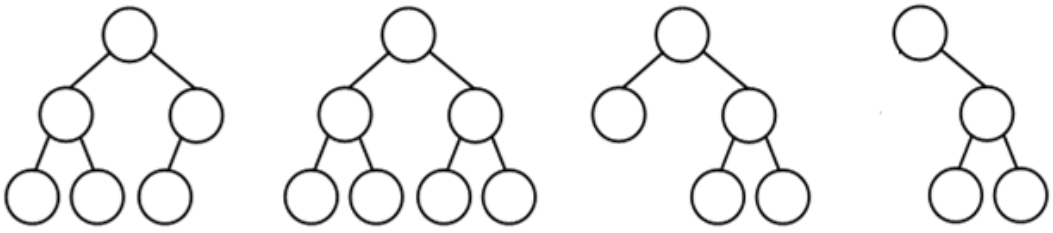
12. Для дерева, изображенного на рисунке, укажите степень вершины B



13. Дерево называется бинарным, если

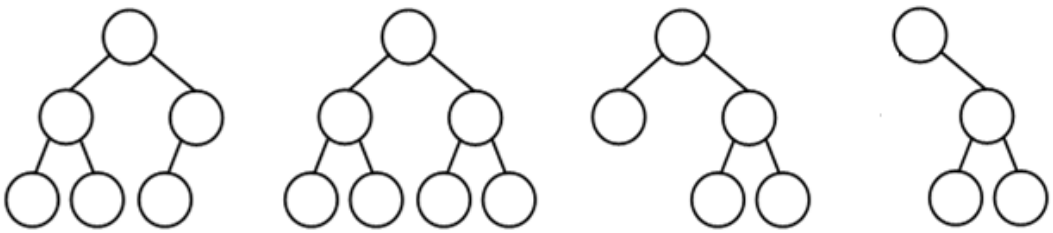
1) каждый узел имеет не более двух потомков
2) каждый узел имеет ровно двух потомков
3) от корня до листа не более двух уровней
4) от корня до листа не менее двух уровней

14. Какие из бинарных деревьев являются строгими?



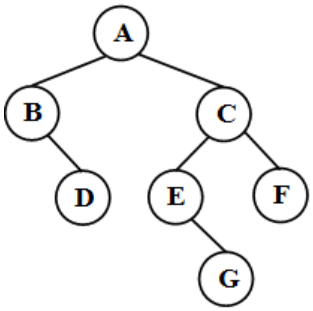
1) 2) 3) 4)

15. Какое из бинарных деревьев являются полным?



1) 2) 3) 4)

16. Укажите вид обхода дерева, представленного на рисунке, если порядок просмотра вершин следующий: B D A E G C F



1) прямой
2) обратный
3) симметричный
4) этот путь не является обходом

17. В чем суть правила обхода дерева в обратном направлении?

1) сначала обрабатывается левое поддерево, потом – правое поддерево, потом – корень поддерева
2) сначала обрабатывается левое поддерево, потом – корень поддерева, потом – правое поддерево
3) сначала обрабатывается правое поддерево, потом – левое поддерево, потом – корень поддерева
4) сначала обрабатывается корень поддерева, потом – правое поддерево, потом – левое поддерево

18. В двоичное дерево поиска были добавлены следующие элементы: 8, 3, 6, 10, 1, 7, 9, 14, 4, 13, 5. Сколько листьев у данного дерева?

19. В двоичное дерево поиска были добавлены следующие элементы: 8, 3, 6, 10, 1, 7, 9, 14, 4, 13, 5. Какова высота дерева?

Критерии оценки:

За каждое правильно выполненное задание назначается 1 балл. В случае задания с выбором нескольких правильных ответов 1 балл дается за полностью выполненное задание, 0 баллов – за хотя бы один неверный ответ. Затем первичные баллы (максимум – 19 баллов) пропорционально переводятся в тестовые баллы согласно технологической карте дисциплины.

Комплект тестовых заданий № 10
по дисциплине «Программирование»

Раздел 11. ***Обобщенное программирование. Коллекции. Linq.***

1. Операции упаковки и распаковки (boxing, unboxing)

1) преобразуют объекты значимого типа в ссылочный и обратно
2) обе операции выполняются автоматически, не требуя явного задания преобразования типов
3) задают преобразование строк из кодировки Unicode в кодировку ASCII и обратно

2. Выберите утверждения, верные для обобщений

1) обобщения позволяют решить проблему безопасности типов
2) использование типов значений с обобщенными классами коллекций вызывает упаковку и распаковку при преобразовании в ссылочный тип и обратно
3) обобщения могут использовать только один универсальный параметр одновременно
4) обобщения повышают степень повторного использования кода

3. Выберите верный оператор для присвоения переменной универсального параметра некоторого начального значения

1) T parameter = 0;
2) T parameter = default(T);
3) T parameter = null;

4. Что будет выведено на экран после выполнения следующего фрагмента кода:

```
List<int> list = new List<int>();
for (int i = 1; i < 4; i++)
   list.Add(i);
list.Insert(2, 5);
list.Insert(1, 6);
Console.WriteLine(list[2]);
```

5. Что будет выведено на экран после выполнения следующего фрагмента кода:

```
List<int> list = new List<int>();
for (int i = 1; i < 10; i++)
    list.Add(i);
list.RemoveAt(1);
list.Remove(3); ;
Console.WriteLine(list[3]);
```

6. Изучите следующий код:

```
Stack<int> stack = new Stack<int>();
stack.Push(0);
stack.Push(1);
stack.Pop();
stack.Push(2);
stack.Pop();
```

Какой элемент остался в стеке после выполнения этого кода?

1) 0
2) 1
3) 2

7. Что будет выведено на экран после выполнения следующего фрагмента кода:

```
Stack<int> stack = new Stack<int>();
for (int i = 0; i < 5; i++)
    stack.Push(2 * i);
for (int i = 0; i < 3; i++)
    stack.Pop();
stack.Push(11);
stack.Push(12);
foreach (int elem in stack)
    Console.Write("{0} ", elem);
```

1) 6 8 11 12
2) 12 11 2 0
3) 0 2 11 12
4) 12 11 6 8

8. Изучите следующий код:

```
Queue<int> queue = new Queue<int>();
queue.Enqueue(0);
queue.Enqueue(1);
queue.Dequeue();
queue.Enqueue(2);
queue.Dequeue();
```

Какой элемент остался в очереди после выполнения этого кода?

1) 0
2) 1
3) 2

9. Что будет выведено на экран после выполнения следующего фрагмента кода:

```
Queue<int> queue = new Queue<int>();
for (int i = 0; i < 5; i++)
    queue.Enqueue(2 + i);
for (int i = 0; i < 3; i++)
    queue.Dequeue();
queue.Enqueue(1);
queue.Enqueue(2);
foreach (int elem in queue)
    Console.Write("{0} ", elem);
```

1) 2 3 1 2
2) 2 1 6 5
3) 2 1 2 3
4) 5 6 1 2

10. Какие элементы будут храниться во множестве после выполнения следующего фрагмента кода:

```
HashSet<int> set = new HashSet<int>();
for (int i = 1; i <= 10; i++)
    set.Add(i % 4);
```

1) 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
2) 1, 2, 3, 0, 1, 2, 3, 0, 1, 2
3) 1, 2, 3, 0

11. Что будет выведено на экран после выполнения следующего фрагмента кода:

```
SortedSet<int> set = new SortedSet<int>();
for (int i = 1; i <= 8; i++)
    set.Add(i % 5);
foreach (int elem in set)
    Console.Write("{0} ", elem);
```

1) 0 1 2 3 4
2) 1 2 3 4 0
3) 1 2 3 4 0 1 2 3
4) 0 1 1 2 2 3 3 4
5) 1 2 3 4 5 6 7 8

12. Что вы можете сказать о словаре Dictionary?

1) словарь позволяет эффективно проверить, содержит ли он ключ
2) словарь позволяет эффективно проверить, содержит ли он значения
3) для каждого ключа словарь хранит только одно значение
4) по разным ключам словарь не может хранить одно и то же значение

13. После выполнения следующего фрагмента кода

```
Dictionary<int, string> dictionary = new Dictionary<int, string>();
dictionary.Add(1, "gold medal");
dictionary.Add(2, "silver medal");
dictionary.Add(3, "bronze medal");
```

```
dictionary.Add(3, "other");

foreach (KeyValuePair<int, string> pair in dictionary)
    Console.WriteLine("{0} = {1}", pair.Key, pair.Value);
```

1) будет сгенерировано исключение ArgumentException
2) на экран будет выведено
   1 = gold medal
   2 = silver medal
   3 = bronze medal
3) на экран будет выведено
   1 = gold medal
   2 = silver medal
   3 = other
4) на экран будет выведено
   1 = gold medal
   2 = silver medal
   3 = bronze medal
   3 = other

14. После выполнения следующего фрагмента кода

```
Dictionary<int, string> dictionary = new Dictionary<int, string>();
dictionary.Add(1, "gold medal");
dictionary.Add(2, "silver medal");
dictionary.Add(3, "bronze medal");
dictionary.Add(4, "no medal");
dictionary[1] = "brilliant medal";

foreach (KeyValuePair<int, string> pair in dictionary)
    Console.WriteLine("{0} = {1}", pair.Key, pair.Value);
```

1) будет сгенерировано исключение ArgumentException
2) на экран будет выведено
   1 = gold medal
   2 = silver medal
   3 = bronze medal
   4 = no medal
3) на экран будет выведено
   1 = brilliant medal
   2 = silver medal
   3 = bronze medal
   4 = no medal
4) на экран будет выведено
   1 = gold medal
   2 = brilliant medal
   3 = bronze medal
   4 = no medal

15. Создана коллекция элементов

```
ArrayList aList = new ArrayList();
aList.Add("Some string value");
aList.Add(700);
aList.Add(5.7);
```

Какой программный код необходимо написать для ее просмотра?

1)
```
foreach (object obj in aList)
{
    Console.WriteLine(obj);
}
```
2)
```
foreach (string s in aList)
{
    Console.WriteLine(s);
}
```
3)
```
foreach (int i in aList)
{
    Console.WriteLine(i);
```

```
    }
4)  foreach (int i in aList)
    {
        Console.WriteLine(aList[i]);
    }
```

Критерии оценки:

За каждое правильно выполненное задание назначается 1 балл. В случае задания с выбором нескольких правильных ответов 1 балл дается за полностью выполненное задание, 0 баллов – за хотя бы один неверный ответ. Затем первичные баллы (максимум – 15 баллов) пропорционально переводятся в тестовые баллы согласно технологической карте дисциплины.

Комплект тестовых заданий № 11
по дисциплине «Программирование»

Раздел 12. ***Работа с файлами и файловой системой.***

1. Какие из перечисленных классов позволяют работать с двоичными файлами?

1) StreamWriter
2) StreamReader
3) FileStream
4) StringWriter
5) BinaryWriter

2. Какой из методов потока записи позволяет записать массив байтов в двоичный файл?

1) Write
2) WriteByte
3) WriteBytes
4) WriteLine

3. Какой из перечисленных классов позволяет записать с помощью метода `WriteLine` в текстовый файл всю строку (не побайтово)?

1) FileStream
2) StreamWriter
3) StringWriter

4. Если файл с именем `info.dat` не существует, что произойдет при попытке выполнения следующей программной строки:
```
FileStream input=new FileStream(”info.dat”,FileMode.Open)
```

1) создается и открывается новый файл с именем info.dat
2) появляется окно каталогов, в котором пользователь должен найти данный файл
3) генерируется исключение FileNotFoundException
4) генерируется исключение IOException

5. Если файл с именем `info.dat` существует, что произойдет при попытке выполнения следующей программной строки:
```
FileStream input=new FileStream(”info.dat”,FileMode.Create)
```

1) создается новый файл с таким именем, старый разрушается
2) открывается существующий файл, все данные будут записываться в конец файла
3) генерируется исключение FileNotFoundException
4) генерируется исключение IOException

6. Для чтения символов из файла используется метод `Read`. Этот метод возвращает:

1) строку (из которой потом извлекаются символы) или null, если больше символов нет
2) очередной символ или null, если больше символов нет
3) код символа или -1, если больше символов нет
4) код символа или 0, если больше символов нет

7. Для чтения символов из файла используется метод `ReadLine`. Этот метод возвращает:

1) строку (из которой потом извлекаются символы) или null, если больше символов нет
2) очередной символ или null, если больше символов нет
3) код символа или -1, если больше символов нет
4) код символа или 0, если больше символов нет

8. При обращении к конструктору `StreamReader("t.txt")` файл `t.txt` должен находиться:

1) в корневом каталоге
2) в каталоге bin/debug текущего проекта

9. Обращение к конструктору `StreamWriter("t.txt", true)` означает, что файл t.txt открывается для:

1) чтения
2) дозаписи
3) перезаписи

10. При достижении конца символьного потока метод `ReadLine()` вернет значение:

1) -1
2) 0
3) null
4) true

11. Произвольный доступ к потоку `FileStream` осуществляется через метод:

1) Read
2) ReadByte
3) Seek
4) Flush

12. Метод `Length` при обращении к потоку `FileStream` определит количество:

1) бит в потоке
2) байт в потоке
3) значений некоторого типа в потоке

Критерии оценки:

За каждое правильно выполненное задание назначается 1 балл. В случае задания с выбором нескольких правильных ответов 1 балл дается за полностью выполненное задание, 0 баллов – за хотя бы один неверный ответ. Затем первичные баллы (максимум – 12 баллов) пропорционально переводятся в тестовые баллы согласно технологической карте дисциплины.

Комплект тестовых заданий № 12
по дисциплине «Программирование»

Раздел 13. ***Визуальное программирование.***

1. Что представляет собой форма в Microsoft Visual Studio?

1) будущее окно приложения, на котором будут размещаться элементы управления
2) окно, в котором приведены все основные свойства выделенного элемента
3) главный компонент Microsoft Visual Studio
4) одно из важных свойств представления

2. Элемент управления – это

1) объект, с помощью которого программа запускается на выполнение
2) объект, являющийся элементом графического интерфейса приложения и реагирующий на события
3) инструмент, который используется для создания объектов на форме
4) объект, появляющийся на экране при запуске программы и предназначенный для остановки программы

3. Элемент управления Label используется для

1) ввода текста с клавиатуры
2) прорисовки фигур на форме
3) отображения текста на форме
4) редактирования текста
5) удаления объектов

4. Для ввода и редактирования текста можно использовать элемент управления

1) Button
2) CheckBox
3) RadioButton
4) TextBox

5. К какому классу принадлежит объект RadioButton?

1) System.Windows.Forms.RadioButton
2) System.Windows.RadioButton
3) System.Forms.RadioButton
4) Forms.RadioButton

6. Элемент управления RadioButton предназначен

1) для ввода и редактирования текста
2) для установки нескольких переключателей одновременно
3) для установки только одного переключателя
4) для размещения изображения на форме
5) для удаления объекта

7. Свойство Text объекта Form определяет

1) цвет фона формы
2) имя формы
3) ширину и высоту формы
4) строку заголовка формы
5) способ размещения формы на экране

8. В программе при обращении к объекту указывается свойство

1) Text
2) Name
3) Size
4) Cursor
5) Font

9. Как изменить надпись на метке label1 на «ABC» в Windows Forms?

1) this.label1 = "ABC";
2) this.label1.Caption = "ABC";

3) this.label1.String = "ABC";
4) this.label1.Text = "ABC";

10. Событие Click происходит

1) при одинарном щелчке мыши на объекте
2) при двойном щелчке мыши на объекте
3) при перемещении мыши по объекту
4) при получении объектом фокуса
5) при изменении содержимого объекта

11. Какой код необходимо написать, чтобы изменить цвет самой формы?

1) this.ForeColor = Color.<цвет>;
2) this.Color = ForeColor.<цвет>;
3) this.BackColor = Color.<цвет>;
4) this.BackColor.<цвет>;

12. Какое свойство меняет цвет текста у текстового поля?

1) ForeColor
2) BackColor
3) Color
4) AutoColor

13. Какой класс содержит методы для построения дуг, отрезков и других геометрических примитивов?

1) Image
2) Graphics
3) Bitmap

14. Какие команды используются для получения красного цвета?

1) Color.Red
2) Color.FromArgb(1.0, 0, 0)
3) Color.FromArgb(255, 0, 0)
4) Color.FromArgb(0, 255, 0, 0)
5) Color.FromArgb(0, 255, 255)

15. Какое из свойств компонента Chart содержит коллекции с отображаемыми данными?

1) Data
2) Sequence
3) Series

Критерии оценки:

За каждое правильно выполненное задание назначается 1 балл. В случае задания с выбором нескольких правильных ответов 1 балл дается за полностью выполненное задание, 0 баллов – за хотя бы один неверный ответ. Затем первичные баллы (максимум – 15 баллов) пропорционально переводятся в тестовые баллы согласно технологической карте дисциплины.

Вопросы к экзамену (1 семестр)
по дисциплине «Программирование»

1. Языки программирования и их классификация.
2. Парадигмы программирования.
3. Структура программы на языке C#.
4. Состав языка C# (алфавит, лексемы, идентификаторы, ключевые слова, литералы, комментарии).
5. Типы данных C#. Встроенные типы. Типы значений и ссылочные типы.
6. Переменные (правила описания, инициализация, область видимости).
7. Преобразование базовых типов данных.
8. Понятие выражения. Типы операций в C#. Приоритет операций.
9. Арифметические операции. Операции инкремента и декремента. Операции присваивания.
10. Логические операции и операции отношения. Условная операция.
11. Операции с разрядами. Операции сдвига.
12. Организация консольного ввода-вывода данных в C#. Форматный вывод.
13. Операторы ветвления.
14. Операторы цикла.
15. Операторы передачи управления.
16. Общие принципы организации массивов. Одномерные массивы: объявление, инициализация.
17. Типовые задачи на обработку одномерных массивов.
18. Двумерные массивы: объявление, инициализация. Массивы трех и более измерений.
19. Ступенчатые массивы.
20. Типовые задачи на обработку двумерных массивов.
21. Класс String. Создание строк. Методы для работы со строками.
22. Класс StringBuilder. Методы для работы со строками.
23. Форматирование строк.
24. Методы-процедуры и методы-функции.
25. Передача параметров по ссылке и по значению. Выходные параметры.
26. Массивы в качестве параметров методов.
27. Методы с переменным числом параметров. Параметры по умолчанию.
28. Рекурсивные методы.
29. Общая форма определения класса. Создание объектов класса.
30. Конструкторы по умолчанию. Создание конструкторов. Ключевое слово this.
31. Статические члены класса. Статические классы.
32. Модификаторы доступа.
33. Инкапсуляция. Свойства.
34. Массивы объектов. Индексаторы.
35. Перегрузка методов и конструкторов.
36. Перегрузка операторов.
37. Основы наследования.
38. Конструкторы и наследование.
39. Полиморфизм и переопределение методов.
40. Абстрактные классы: определение, применение.
41. Класс System.Object и его методы.
42. Интерфейсы: определение, применение.
43. Стандартные интерфейсы .NET. Сравнение объектов.
44. Определение структуры. Создание объектов структуры. Отличия структур от классов.
45. Структура DateTime. Форматирование дат и времени. Операции со структурой DateTime.
46. Перечисления: синтаксис объявления, особенности использования.

Критерии оценки:

– оценка «отлично» выставляется, если обучающийся в полном объеме усвоил материал программы учебной дисциплины, раскрыл теоретическое содержание вопросов билета, не затрудняется в ответах на дополнительные вопросы экзаменатора, успешно выполнил практическое задание, продемонстрировав необходимые навыки и умения правильного применения теоретических знаний в практической деятельности;

– оценка «хорошо» выставляется, если обучающийся знает материал программы учебной дисциплины, правильно, по существу и последовательно излагает содержание вопросов билета, в целом правильно выполнил практическое задание, владеет основными умениями и навыками, при ответе не допустил существенных ошибок и неточностей;

– оценка «удовлетворительно» выставляется, если обучающийся усвоил только основные положения материала программы учебной дисциплины, содержание вопросов билета изложил поверхностно, без должного обоснования, допускает неточности и ошибки, недостаточно правильные формулировки, нарушает последовательность в изложении материала, практическое задание выполнил не в полном объеме, испытывает затруднения при ответе на часть дополнительных вопросов;

– оценка «неудовлетворительно» выставляется, если обучающийся не знает основных положений материала программы учебной дисциплины, при ответе на вопросы билета допускает существенные ошибки, не выполнил практическое задание, не смог ответить на большинство дополнительных вопросов или отказался отвечать.

Вопросы к экзамену (2 семестр)
по дисциплине «Программирование»

1. Обработка исключений. Конструкция try..catch..finally.
2. Типы исключений. Класс Exception. Обработка многочисленных исключений.
3. Вложенные конструкции try..catch.
4. Генерация исключения и оператор throw.
5. Пользовательские классы исключений.
6. Определение делегатов. Соответствие методов делегату. Добавление методов в делегат.
7. Анонимные методы.
8. Лямбда-выражения. Лямбда-выражения как параметры методов.
9. Делегаты Action, Predicate и Func.
10. Создание и вызов событий. Добавление и удаление обработчиков событий.
11. Понятие структуры данных. Классификация структур данных. Основные структуры данных.
12. Односвязный список. Программная реализация односвязного списка.
13. Двусвязный список. Программная реализация двусвязного списка.
14. Стек. Программная реализация стека.
15. Очередь. Дек. Программная реализация очереди.
16. Деревья. Бинарные деревья. Двоичное дерево поиска.
17. Программная реализация двоичного дерева поиска.
18. Понятие обобщения. Преимущества применения обобщений.
19. Обобщенные классы. Статические поля обобщенных классов.
20. Обобщенные методы.
21. Обобщенные делегаты.
22. Обобщенные интерфейсы.
23. Ограничения обобщений.
24. Сравнение экземпляров обобщенных типов.
25. Необобщенные коллекции. Обобщенные коллекции.
26. Списки: классы List<T> и LinkedList<T>.
27. Очередь: класс Queue<T>.
28. Стек: класс Stack<T>.
29. Словари: классы Dictionary<TKey, TValue> и SortedDictionary<TKey, TValue>.
30. Множества: классы HashSet<T> и SortedSet<T>.
31. Основы LINQ. Операторы запросов LINQ. Методы расширения LINQ.
32. LINQ. Фильтрация данных. Проекция данных.
33. LINQ. Сортировка данных.
34. LINQ. Агрегатные операции.
35. LINQ. Проверка наличия и получение элементов.
36. LINQ. Получение части коллекции.
37. Виды файлов. Потоки данных. Классы .NET для работы с файлами.
38. Чтение и запись файлов. Класс FileStream.
39. Чтение и запись текстовых файлов. Классы StreamReader и StreamWriter.
40. Работа с файловой системой. Классы Directory и DirectoryInfo.
41. Работа с файлами. Классы File и FileInfo.
42. Создание приложения с графическим интерфейсом. Основные свойства форм и элементов управления.
43. Делегаты и события. Делегат System.EventHandler. Обработка событий в Windows Forms. События формы.
44. Диалоговые окна в Windows Forms. Модальные и немодальные диалоговые окна. Стандартные диалоговые окна.
45. Разработка многооконных приложений Windows Forms. Взаимодействие форм. Передача информации между формами.
46. Работа с графическими файлами. Класс Image. Класс Bitmap.
47. Класс Graphics. Методы и свойства класса Graphics.
48. Инструменты для рисования. Перья. Кисти.

Критерии оценки:

– оценка «отлично» выставляется, если обучающийся в полном объеме усвоил материал программы учебной дисциплины, раскрыл теоретическое содержание вопросов билета, не затрудняется в ответах на дополнительные вопросы экзаменатора, успешно выполнил практическое задание, продемонстрировав необходимые навыки и умения правильного применения теоретических знаний в практической деятельности;

– оценка «хорошо» выставляется, если обучающийся знает материал программы учебной дисциплины, правильно, по существу и последовательно излагает содержание вопросов билета, в целом правильно выполнил практическое задание, владеет основными умениями и навыками, при ответе не допустил существенных ошибок и неточностей;

– оценка «удовлетворительно» выставляется, если обучающийся усвоил только основные положения материала программы учебной дисциплины, содержание вопросов билета изложил поверхностно, без должного обоснования, допускает неточности и ошибки, недостаточно правильные формулировки, нарушает последовательность в

изложении материала, практическое задание выполнил не в полном объеме, испытывает затруднения при ответе на часть дополнительных вопросов;
– оценка «неудовлетворительно» выставляется, если обучающийся не знает основных положений материала программы учебной дисциплины, при ответе на вопросы билета допускает существенные ошибки, не выполнил практическое задание, не смог ответить на большинство дополнительных вопросов или отказался отвечать.