ЧАСТ 1 (Време за работа: 90 минути)

Отговорите на задачите от 1. до 16. включително отбелязвайте в листа за отговори!

1. Какъв ще е резултатът от изпълнението на следния програмен фрагмент?

```
int a = 2, b = 3, c = 4;
switch (a * b – c % b)
{
   case 0: case 2: case 4:
      ++a;
      c += b;
      b -= a;
      break;
   case 1: case 3: case 5:
      --a;
      c -= b;
      break;
   default:
      a += b;
      break;
}
Console.WriteLine(a + " " + b + " " + c);
```

А) 1 3 1 Б) 3 7 3 В) 3 7 0 Г) 5 3 4

2. На кой етап от създаване на информационната система се проверява системата в симулирана среда?

А) анализ Б) тестване В) експлоатация и поддръжка Г) разработка

3. Кое от изброените не се поддържа в C#?

А) клас може да наследи един или няколко класа

Б) интерфейс може да наследи един или няколко класа

В) клас може да наследи няколко класа и няколко интерфейса

Г) нито едно от изброените не се поддържа

4. Какво ще изведе програмният фрагмент след изпълнението си?

```
int x = 4, y, z;
y = --x;
z = x--;
Console.WriteLine($"{x} {y} {z}");
```

А) 3 2 3 Б) 2 3 3 В) 3 2 2 Г) 2 3 4

5. Даден е текстов файл fruits.txt, в който е записан текстът “apple banana lemon kiwi grape melon tomato pear cherry peach apricot lime orange” (всяка дума е на нов ред).

Какво ще се изведе след изпълнение на програмния фрагмент?

```
StreamReader reader = new StreamReader("fruits.txt");
string fruit = reader.ReadLine();
int line = 1;
int c = 1;
while (fruit != null)
{
   if (line % 3 == 0)
   {
      Console.WriteLine($"{c++}. {fruit}");
   }
   fruit = reader.ReadLine();
   line++;

}
reader.Close();
```

А)
2. lemon
3. melon
4. cherry
5. lime

Б)
1. kiwi
2. tomato
3. peach
4. orange

В)
1. apple
2. kiwi
3. tomato
4. peach
5. orange

Г)
1. lemon
2. melon
3. cherry
4. lime

6. Кое от следните твърдения НЕ е вярно за абстрактен клас?

А) неабстрактен клас, наследяващ абстрактен клас, трябва да включва реализации на всички наследени абстрактни методи

Б) абстрактния клас може да съдържа абстрактни методи

В) абстрактния клас може да се инициализира

Г) абстрактният клас се дефинира със запазената дума abstract

7. Коя структура на данни в C# ви позволява да съхранявате двойки ключ-стойност и често се използва за бързо извличане на данни?

А) ArrayList Б) List В) Array Г) Dictionary

8. За какво се използват полетата в C# класовете?

А) Да се определят поведенията на класа

Б) Да съхраняват информация и данни, съдържащи се в обекта на класа

В) За комуникация между класове и обект

Г) За съхраняване на стойността на дефиницията на класа

9. Какъв ще е резултатът от изпълнението на дадения код?

```
public class A
{
   public int i;
   private int j;
}
public class B : A
{
   public void display()
   {
      base.j = base.i + 1;
      Console.WriteLine(base.i + "  " + base.j);
   }
}
internal class Program
{
   static void Main(string[] args)
   {
      B obj = new B();
      obj.i = 1;
      obj.j = 2;
      obj.display();

   }
}
```

А) 1 3 Б) 2 3 В) 1 2 Г) грешка по време на компилиране

10. Каква е грешката в дадения код?

```
static void MyFunction()
{
   {
      int a = 10;
      int b = 20;
      int c = a + b;
   }
   Console.WriteLine(c);
}
```

А) Променлива **c** вече не съществува извън блока.

Б) Променливите **a** и **b** никога не се използват.

В) Не можете да поставите код в скоби в друг блок.

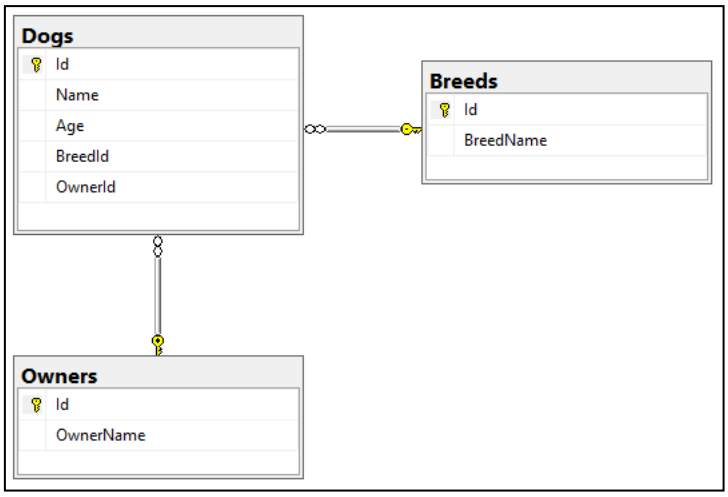
Г) Променливата **c** никога не се използва; показването ѝ на конзолата не се брои за използване.

11. Кой от регулярните изрази ще прихване C# идентификаторите в даден текст (идентификаторът в C# може да съдържа букви, цифри и долна черта и не може да започва с цифра)

А) \b[a-zA-Z_][a-zA-Z0-9_]?\b Б) \b[a-zA-Z_][a-zA-Z0-9_]*\b

В) [a-zA-Z_][a-zA-Z0-9_]* Г) \b[a-zA-Z_][a-zA-Z0-9_]{1,}

12. На диаграмата са представени таблици, свързани с кучета и техните стопани. Кое от твърденията за отношенията между таблиците е вярно?



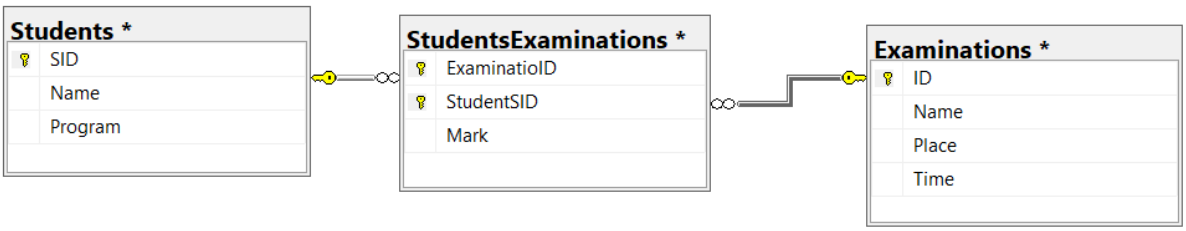
А) Връзката между таблиците **Dogs** и **Owners** е едно към едно

Б) Връзката между таблиците **Breeds** и **Dogs** е много към много

В) **BreedId** и **OwnerId** в таблицата Dogs са външни ключове

Г) Връзката между таблиците **Dogs** и **Owners** е много към много

13. Дадена е база данни със следната диаграма



Кои от следните SQL изрази връщат само различните имена (поле Name) в таблица Students?

1) SELECT DIFFERENT Name FROM Students;
2) SELECT DISTINCT Name FROM Students;
3) SELECT Name FROM STUDENTS GROUP BY Name;
4) SELECT UNIQUE Name FROM Students;
5) SELECT t.Name FROM Students INNER JOIN Students AS t ON Students.Name = t.Name;

А) 2 и 3 Б) 2, 3 и 4 В) 2, 3 и 5 Г) 1, 2 и 4

14. Кои програмни фрагменти ще извлекат от текста “C# is a programming language” фрагмента “C# language”?

1	`string text = "C# is a programming language";`
	`int position = text.IndexOf('l');`
	`text = text.Substring(0, 3) + text.Substring(position);`
	`Console.WriteLine(text);`

2	`string text = "C# is a programming language";`
	`text = text.Substring(0, 2) + text.Substring(20);`
	`Console.WriteLine(text);`

3	`string text = "C# is a programming language";`
	`int index1 = text.IndexOf(' ');`
	`int index2 = text.LastIndexOf('l');`
	`text = text.Remove(index1 + 1, index2 - index1 - 1);`
	`Console.WriteLine(text);`

4	`string text = "C# is a programming language";`
	`int index1 = text.IndexOf(' ');`
	`int index2 = text.LastIndexOf(' ');`
	`text = text.Replace(text.Substring(index1, index2 - index1), "");`
	`Console.WriteLine(text);`

А) 1 и 4 Б) 1, 3 и 4 В) 2 и 4 Г) 1 и 3

15. Какво ще изведе следния програмен фрагмент?

```
int[] a = { 1, 2, 4, 9 };
int[] b = { 2, 6, 7, 12, 15, 21 };
int[] c = new int[a.Length + b.Length];

int p = a.Length - 1, q = b.Length - 1, k = 0;

while (p >= 0 && q >= 0)
{
   if (a[p] > b[q]) c[k++] = a[p--];
   else c[k++] = b[q--];
}

for (int i = p; i >= 0; i--)
{
   c[k++] = a[i];
}

for (int i = q; i >= 0; i--)
{
   c[k++] = b[i];
}


Console.WriteLine(string.Join(", ", c));
```

А) 21, 15, 12, 9, 7, 6, 4, 2, 2, 1

Б) 1, 2, 2, 6, 4, 7, 9, 12, 15, 21

В) 1, 2, 2, 4, 6, 7, 9, 12, 15, 21

Г) 1, 7, 9, 15, 21, 2, 2, 4, 6, 12

16. Коя от показаните редици от числа ще се изведе на конзолата?

```
static void Main(string[] args)
{
   HashSet<int> hs = new HashSet<int>() { 1, 1, 2, 3, 3, 4 };
   hs.Remove(hs.Count);
   Console.WriteLine(string.Join(", ", hs));
}
```

А) 1, 1, 2, 3, 3 Б) 1, 2, 3, 4 В) 1, 2, 3 Г) 1, 1, 2, 3, 3, 4

Отговорите на задачите от 17. до 24. вкл. запишете в листа за отговори!

17. Какво ще се изведе на конзолата като резултат от изпълнението на програмата?

```
static int Recursive(int value, ref int count)
{
   count++;
   if (value >= 10)
   {
      return value;
   }
   return Recursive(value + 1, ref count);
}

static void Main(string[] args)
{
   int count = 0;
   int total = Recursive(5, ref count);

   Console.WriteLine(total);
   Console.WriteLine(count);
}
```

18. Какво ще се изведе на стандартния изход след изпълнението на следния програмен фрагмент?

```
try
{
            int[] a = { 1, 2, 3, 4 };
            for (int i = 0; i < 5; ++i)
            {
                        Console.Write(a[i]);
   }
   Console.WriteLine();
   int x = 1 / Convert.ToInt32(0);
}
catch (IndexOutOfRangeException e)
{
   Console.WriteLine("The index is out of range.");
}
catch(ArithmeticException e)
{
   Console.WriteLine("Division by zero.");
}
```

19. В таблицата са изброени елементи на графичния интерфейс и е описано предназначението им. Свържете всеки елемент с неговото предназначение, като в листа за отговори срещу числата 1, 2, 3 и 4 запишете съответната буква.

А. button	1.избор на една стойност от група с краен брой допустими стойности
Б. panel	2. въвеждане на данни
В. comboBox	3. предизвикване на действия
Г. textBox	4. поставяне на група от свързани компоненти

20. Дадена е таблицата cars със следните данни:

Id	FirstName	LastName	Age
1	Ivan	Yordanov	25
2	Peter	Johnson	18
3	Mark	Peterson	35
4	Yordan	Yordanov	30
5	George	Jones	29

А) Напишете заявка, която променя възрастта на Ivan на 24 години.

Б) Напишете заявка, която изтрива от таблицата цялата информация за хората на възраст над 30 години.

21. Дадени са таблиците Animals и Teritories със следните данни:

Id	Type	Age	TeritoryId
1	Lion	5	1
2	Elephant	10	2
3	Penguin	2	3
4	Tiger	4	1

Id	Name	Area	Continent
1	Savannah	5000	Africa
2	Jungle	8000	Asia
3	Iceberg	2000	Antarctica

```
SELECT a.Type, a.Age, t.Name
FROM Animals AS a
JOIN Teritories AS t
ON a.TeritoryId = t.Id
WHERE t.Name = 'Savannah'
```

В листа за отговори запишете колко реда и колко колони ще има таблицата, получена в резултат от изпълнението на горната заявка?

22. Дадена е програма, която намира сбора на цифрите в стринг. В кода има допуснати грешка. В листа за отговори напишете верния вариант код.

```
string sentence = "Your birthday is April 12, 1998";
for (int i = 0; i < sentence.Length; i++)
{
  int sum = 0;
  if (senrense[i].IsDigit()) sum += (sentence[i] – 0);
}
Console.WriteLine(sum);
```

23. Какво ще изведе дадения програмен фрагмент?

```
internal class Program
{

  static void f(Queue<int> Q)
  {
    int i;
    if (Q.Count()>0)
    {
      i = Q.Dequeue();
      f(Q);
      Q.Enqueue(i);
    }
  }

  static void Main(string[] args)
  {
    Queue<int> Q = new Queue<int>();
    Q.Enqueue(1);
    Q.Enqueue(2);
    Q.Enqueue(3);
    Q.Enqueue(4);
    Q.Enqueue(5);
    f(Q);
    Console.WriteLine(string.Join(" ", Q));
  }
```

24. Даден е програмен фрагмент, който намира сбора на елементите по колони в квадратна таблица от числа. В кода са пропуснати части, които са означени с (1), (2) и (3). В листа за отговори срещу числата 1 и 2 запишете пропуснатия код, а срещу 3 запишете какво ще изведе последната команда в дадения код.

```
int[,] a = new int[4, 4];

for (int i = 0; i < 4; i++)
{
   for (int j = 0; j < 4; j++)
   {
      a[i, j] = i + j;
   }
}
for (int i = 0; i < 4; i++)
{
   (1)
   for (int j = 0; j < 4; j++)
   {
      (2)
   }
   Console.WriteLine(sum);
}
```

ЧАСТ 2 (Време за работа: 150 минути)

Файловете с отговорите на задачите от 25. до 28. включително, запишете в изпитната система като спазите указанията в условието на задачата!

Внимание! Имената на работните файлове, които прикачвате в изпитната система НЕ трябва да съдържат текстове или символи, които могат да доведат до нарушаване на анонимността на изпитната Ви работа!

25. Създайте проект с име **zad25**, в който се прочита от стандартния вход естествено число n, 1 ≤ n ≤ 10000000. Програмата проверява и извежда на стандартния изход съобщението: „<число> is a mirror prime“, ако въведеното число е „огледално просто“. Ако въведеното число не е просто да се извежда съобщението „<число> is NOT a prime“. Ако въведеното число е просто но огледалното му число не е, то изведете съобщението: „<число> is NOT a mirror prime“. Направете необходимата валидация и обработка на изключение. Ако въведената данна не е валидна да се извежда съобщението „Incorrectly entered number“.

Упътване: **Огледално просто** ще наричаме естествено число, което е просто и неговото число-палиндром също е просто (напр 97 и 79).

**Забележка*: *Приемат се и решения с графичен потребителски интерфейс (ГПИ), в които числото n се въвежда в текстово поле, а резултатът се извежда в етикет или в текстово поле, което не може да се редактира.*

Пример

Вход	Изход
97	97 is a mirror prime
45	45 is NOT a prime
29	29 is NOT a mirror prime
eleven	Incorrectly entered number

25 зад. 10 точки

Метод за определяне на простите числа – 3,5 точки

Конструиране на огледалното число – 3,5 точки

Проверка на различните случаи – 1,5 точки

Изключения – 1,5 точки

26. Създайте проект с име zad26. Приложението приема като входни данни група от естествени числа и реализира различни операции с тях. Запишете числата в колекция по ваш избор.
Да се напишат следните методи:

- **Conversion**() – по зададена колекция от числа връща колекция от същия вид, в която всяко число от входната редица се заменя със сбора от цифрите му.
- **Unique**() – по зададена колекция от числа връща сортирана в нарастващ ред колекция от същия вид, в която са премахнати повтарящите се числа от дадената колекция.

- **EvenOddSort**() – по зададена колекция от числа връща колекция от същия вид, в която числата са подредени по следния начин: първо четните, подредени в нарастващ ред, а след това нечетните подредени в намаляващ ред.
- **EvenSimetry**() – по зададена колекция от числа връща колекция от същия вид, в която числата са подредени по следния начин: числата на четни позиции да се разменят със симетрично стоящите спрямо средата на колекцията.
- **SortByDigitSum**() – по зададена колекция от числа връща колекция от същия вид, в която числата са подредени в нарастващ ред по сбора от цифрите си.

Числата ще се въвеждат по едно или повече на ред, разделени с интервал, до въвеждане на команда **END**.
Следва поредица от команди, които извикват описаните методи. След изпълнението на всяка команда базовата редица се преобразува и следващата команда се прилага върху редактираната редица.
След всяка команда се извежда актуалното състояние на редицата във формат {**num1} {num2} {num3} …**
Въвеждането на команди спира при срещане на **STOP.**

Примерен вход	Примерен изход
72 67 128 1 79 144 121 86 13 256 16 85 9 83 169 END EvenOddSort EvenSimetry Conversion Unique SortByDigitSum Conversion Unique STOP	*EvenOddSort* 16 72 86 128 144 256 169 121 85 83 79 67 13 9 1 *EvenSimetry* 1 72 13 128 79 256 85 121 169 83 144 67 86 9 16 *Conversion* 1 9 4 11 16 13 13 4 16 11 9 13 14 9 7 *Unique* 1 4 7 9 11 13 14 16 *SortByDigitSum* 1 11 4 13 14 7 16 9 *Conversion* 1 2 4 4 5 7 7 9 *Unique* 1 2 4 5 7 9

26 зад. 15 точки

Четене на числата и записване в колекцията 1 точка

Метод за намиране сбора от цифрите на число – 2 точки

Conversion – 2 точки

Unique – 2 точки

EvenOddSort – 2 точки

EvenSimetry – 2 точки

SortByDigitSum – 3 точки

Main –1 точка

27. Създайте приложение **zad27** което съхранява информация зя баскетболни отбори и техните състезатели. За целта реализирайте класовете **Player и Team (**спазвайте принципа на енкапсулация)**.**

Класът **Player** има следните полета:

- **Name: string**
- **Position: string**
- **Rating: double**
- **Games: int**

✓ Създайте конструктор, който получава име, позиция, рейтинг, брой игри.

✓ Класът трябва да има пренаписан метод **ToString()**, който извежда информация във формат:

"-Player: {name}

--Position: {position}

--Rating: {rating}

--Games played: {games}"

Създайте клас **Team**, който има:

- **колекция Players** от играчи (тип **Player)**

Team има и следните полета:

- **Name: string**
- **OpenPositions: int**
- **Group: char**

✓ Създайте конструктор, който получава името на отбора, openPositions и групата.

Реализирайте следните функционалности:

✓ метод **Count** - връща броя на играчите в отбора.

✓ метод **string AddPlayer(Player player)** – добавя играч към колекцията на отбора, ако има свободни позиции. Преди да добавите играч, проверете:

- ако **name** или **position** са **null или empty**, върнете **"Invalid player's information.".**
- ако няма повече свободни позиции, върнете "**There are no more open positions.".**
- ако рейтингът е под **80**, върнете **"Invalid player's rating.".**
- във всички останали случаи върнете **"Successfully added {playerName} to the team. Remaining open positions: {openPositions}."**

✓ метод **bool RemovePlayer(string name)** – премахва играч по зададено име.

o Ако такъв съществува, връща **true**;

o В противен случай връща **false**.

Не забравяйте да актуализирате полето **OpenPositions**!

✓ **Report()** – извежда информация за отбора и играчите в следния формат:
"Players competing for Team {team} from Group {group}:
{Player$_1$}
{Player$_2$}
{…}"

Приложението чете входните данни от текстов файл **info.txt.** Погрижете се да прихванете възможните изключения при работа с файлове.

На вход може да получите някоя от следните команди:

- **CT (create team)**

    - създава отбор
    - на същия ред следва информация за отбора (име, свободни позиции и група)
    - данните са разделени с интервал
    - след като създадете отбора изведете съобщение **„Create {name} successfully“**
- **CP (count of players)**
    - връща броя на играчите в отбора
- **AP (add player)**
    - добавя играч към отбора
    - на същия ред следват име на играч, позиция на която играе, рейтинг и брой изиграни мачове
    - данните са разделени с интервал
- **RP (remove player)**
    - премахва играч от отбора
    - на същия ред получавате име на играч
    - данните са разделени с интервал
- **RT (report of team)**
    - извежда информация за отбора

info.txt
CT BHTC 5 A AP Viktor Center 97.5 10 RP Slavi AP Slavi PointGuard 94.3 47 AP Evgeni ShootingGuard 93.7 16 AP Momchil SmallForward 67.9 3 AP Vasil PowerForward 86.9 10 AP Stefan Center 95.6 25 AP Ivan SmallForward 98.5 89 RP Slavi CP RT

Примерен изход
Create BHTC Successfully Successfully added Viktor to the team. Remaining open positions: 4 False Successfully added Slavi to the team. Remaining open positions: 3 Successfully added Evgeni to the team. Remaining open positions: 2 Invalid player's rating. Successfully added Vasil to the team. Remaining open positions: 1 Successfully added Stefan to the team. Remaining open positions: 0

There are no more open positions.
True
4
P**layers** competing for Team BHTC from Group A:
-Player: Viktor
--Position: Center
--Rating: 97.5
--Games played: 10
-Player: Evgeni
--Position: ShootingGuard
--Rating: 93.7
--Games played: 16
-Player: Vasil
--Position: PowerForward
--Rating: 86.9
--Games played: 10
-Player: Stefan
--Position: Center
--Rating: 95.6
--Games played: 25

Клас Player	метод Count 1
Полета 1	метод AddPlayer 3
Конструктор 2	метод RemovePlayer 2
toString 2	метод Report 2
клас Team	четене от текстов файл 2
полета 1	Split и изпълняване на команди 2
конструктор 2	

28. Създайте база данни **DZI**. Базата съдържа таблици Students и Subjects. В нея се съхраняват данни за учениците и профилиращите предмети, които са избрали за 2-ра и 3-та матура. ДЗИ 1 винаги е Български език и литература и всички се явяват на него. ДЗИ 2 също е задължителен, но предмета е по избор. Ученик може да се яви на минимум 2 и максимум 3 матури.

1. Създайте таблиците:

Subjects със следните колони:

- ID – цяло число, първичен ключ, автоматична номерация
- Name – име на предмет, текст, задължително поле

Students със следните колони:

- ID – цяло число, първичен ключ, автоматична номерация
- Name – име на ученик, текст, задължително поле
- Town – има не град, текст, задължително поле
- DZI2 – цяло число, id на матура, задължително поле
- DZI3 – цяло число, id на матура

2. Напишете заявки за въвеждане на следните данни в таблиците

ID	**Name**

1	Английски език
2	Математика
3	Информатика
4	Информационни технологии
5	Биология и ЗО
6	География и икономика
7	История и цивилизации

Таблица **Subjects**

Таблица **Students**

ID	Name	Town	DZI2	DZI3
1	Иван Иванов	София	2	1
2	Петър Петров	Варна	1	
3	Илияна Иванова	Плевен	1	6
4	Елизабет Хаджиева	София	3	1
5	Илиян Петров	София	4	
6	Ивайла Димитрова	Плевен	5	4

3. Напишете заявка, която извежда данните за учениците от София, които ще се явяват и на трети ДЗИ. Покажете името на ученика и имената на предметите, които той е избрал за втора и трета матура. Списъкът да е подреден по име на ученик по азбучен ред.
4. Напишете заявка, която извежда име и град на всички ученици, чиито имена започват с „И“ и ще се явят на 3 матури. Подредете по град по азбучен ред и след това по име на ученик отново по азбучен ред.
5. Напишете заявка която намира броя на учениците, които са избрали трета матура.
6. Напишете заявка, която намира градовете, в които нито един ученик не е избрал матура по информатика.
7. Напишете заявка, която намира броя на учениците от всеки град.

Прикачете в изпитната система zip архив с име **zad28**, в който е файла с написаните от Вас заявки. Името на файла със заявките трябва да бъде **zad28(txt/sql).**

създаване на таблици

Subjects 1

Students 2

INSERT заявки – 2

Заявка 3 – 2

Заявка 4 – 2

Заявка 5 – 2

Заявка 6 – 2

Заявка 7 – 2