Odpovědi pište na zvláštní odpovědní list s vaším jménem a fotografií. Pokud budete odevzdávat více než jeden list s řešením, tak se na 2. a další listy nezapomeňte podepsat. Do zápatí všech listů vždy napište i/N (kde i je číslo listu, N je celkový počet odevzdaných listů).

Společná část pro otázky označené A

Předpokládejte následující program:

```
using var a6 = new A1 { X = 6 };
a6.m();
{
   using var a7 = new A1 { X = 7 };
   a7.m();
}
Console.WriteLine("end");

class A1 : IDisposable {
   public int X { get; init; }
   public void m()
      => Console.WriteLine($"A1.m {X}");
   public void Dispose()
      => Console.WriteLine($"A1.Dispose {X}");
}
```

Otázka č. 1 (A)

Co uvedený program vypíše? Detailně vysvětlete.

Otázka č. 2 (A)

Přepište hlavní program bez použití `using` tak, aby se ale choval stále stejně jako program původní.

Společná část pro otázky označené B

Předpokládejte následující program:

```
struct Point {
   public int X;
   public int SqX { get => X * X; }

   public Point(int x, int y) {
      this.X = x;
   }
   public static void Reset(Point p) {
      p.X = 100;
   }
   public void Update() {
      this.X += 1;
} }
class Prg3 {
   public static void Main() {
      Point p1 = new Point(5, 10);
      p1.Update();
      Console.WriteLine($"X: {p1.X}");

      Point.Reset(p1);
      Console.WriteLine($"X: {p1.X}");
} }
```

Otázka č. 3 (B)

Co vypíše uvedený program na standardní výstup? Detailně vysvětlete proč.

Otázka č. 4 (B)

Detailně vysvětlete, jaká je velikost proměnné `p1` a jakou má v paměti strukturu. Dále detailně vysvětlete, jaká je velikost instance `Point` a jak instance `Point` souvisí se samotnou proměnnou `p1`.

Otázka č. 5

Co níže uvedený program vypíše? Detailně vysvětlete proč. Zároveň detailně vysvětlete, co bude obsahem proměnné o a proč.

```
int x = 13;
object o = x;
var y = (long) o;
Console.WriteLine(y);
```

Společná část pro otázky označené C

Předpokládejte následující program:

```
C6 c6 = new B6();

Console.WriteLine(((I6) c6).f());

Console.WriteLine(((A6) c6).f());

Console.WriteLine(((D6) c6).f());

interface I6 {
   int f();
}

class C6 : I6 {
   public int f() => 1;
}

abstract class A6 : C6, I6 {
   public abstract int f();
}

sealed class B6 : A6 {
   public override int f() => 2;
}

class D6 : C6 {
   public new int f() => 3;
}
```

Otázka č. 6 (C)

Co uvedený program vypíše na 1. řádku výstupu? Detailně vysvětlete proč – jako součást vysvětlení nakreslete interfacovou tabulku metod pro interface `I6` u tříd `A6`, `B6`, `C6`, `D6`.

Otázka č. 7 (C)

Co uvedený program vypíše na 2. řádku výstupu? Detailně vysvětlete proč – jako součást vysvětlení nakreslete VMT tříd `A6`, `B6`, `C6`, `D6`.

Otázka č. 8 (C)

Co uvedený program vypíše na 3. řádku výstupu? Detailně vysvětlete proč – jako součást využijte vysvětlení u předchozí otázky.

Otázka č. 9

Co nejlépe navrhněte a naimplementujte třídu `FileLogger`, která má poskytovat metodu `Log`, která má zalogovat předaný `string` do souboru. Zároveň má metoda brát informaci o důležitosti logované zprávy – zatím chceme podporovat pouze 3 úrovně: `info`, `warning`, `error`. Každé volání `Log` má do souboru připsat 1 řádek v následující podobě (pokud bychom např. chtěli zalogovat zprávu „Něco se stalo" na úrovni `warning`):

```
warning: Něco se stalo.
```

Dále předpokládejte, že do budoucna budeme chtít přidávat i další třídy, které mají být z pohledu kódu vyžadujícího logování záměnné – např. `JsonLogger` – napište kostru implementace této třídy, tj. deklaraci metod/vlastností, které tato třída bude mít (těla jejich metod již ale neimplementujte).

Vysvětlete vhodnost a výhody celého vašeho návrhu.

Otázka č. 10

Pro následující třídu dopište podporu pro logování s využitím vašeho návrhu **z otázky 9** – třída `Character` má zalogovat `info`, pokud se zdraví postavy dostane pod 50%:

```
public record class Character {
   public int MaxHealthPoints { get; set; }
   public double HealthPercentage { get; set; }

   public int HealthPoints {
      get {
         return (int) (
            HealthPercentage * MaxHealthPoints
         );
      }
      set {
         HealthPercentage =
            (double) value / MaxHealthPoints;
      }
   }
}
```

Vysvětlete vhodnost a výhody vašeho návrhu.

VŽDY, KDE $\leq 50\%$