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).

Otázka č. 1

Předpokládejte následující datovou strukturu, která je součástí komplexního systému, jež navrhujeme pro projekční kanceláře železničního zabezpečovacího zařízení, stejně tak i pro využívání udržujícími pracovníky státní organizace SŽDC:

```
public sealed class Node {
    public double Longitude { get; set; }
    public double Latitude { get; set; }
    public TrackSegment[] Segments { get; set; }
}

public sealed class TrackSegment {
    public Node Node0 { get; set; }
    public Node Node1 { get; set; }
    public SortedSet<Element> Elements
        { get; } = new SortedSet<Element>();
}

public abstract class Element {
    public Guid Guid
        { get; set; } = Guid.NewGuid();
    public TrackSegment OnSegment { get; set; }
    public double Location { get; set; }
}

public sealed class Signal : Element { ... }
public sealed class Sign : Element { ... }
public sealed class AxleCounter : Element { ... }
public sealed class TrackCircuitBoundary : Element
{ ... }
```

(A) Třída `TrackSegment` reprezentuje jeden souvislý úsek koleje bez výhybek, resp. část výhybky za/před výměnou. Ke každému takovému segmentu koleje může být přiřazeno libovolné množství prvků odvozených od tříd `Element` (např. návěstidla = `Signal`, senzory počítačů náprav = `AxleCounter`, apod.), kde každý takový prvek je jednoznačně identifikován v rámci celé železniční sítě pomocí svého unikátního ID (instance `Guid`). Dále má prvek přiřazen jednoznačnou polohu v metrech (`Location`) od začátku (od uzlu `Node0`) kolejového segmentu – poloha na segmentu je určena jako průsečík osy koleje s kolmicí na osu koleje procházející místem fyzického umístnění daného elementu. Různé elementy na jednom kolejovém úseku mohou mít stejnou polohu – např. návěstidlo a v jeho úrovni na kolejnici umístněný senzor počítače náprav.
Je v této situaci vhodný návrh a implementace tříd `Element` a jejich potomků vzhledem k jejich vkládání do `SortedSet` položky `Elements`? Pokud ano, tak vysvětlete proč. Pokud ne, tak vysvětlete proč, a třídy vhodně upravte.
[1 bod]

(B) V uvedené aplikaci budeme programovat velké množství algoritmů, které různými způsoby zpracovávají/pracují s výše uvedenou datovou strukturou (např. generování schéma tratě ve formátu SVG, generování zjednodušeného schématu pro zabezpečovací zařízení, generování tabulky povolených cest = tzv. závěrová tabulka, generování rozdělení ovládání jednotlivých prvků do EIP bloků staničního zabezpečovacího zařízení AŽD ESA 33, apod.). Navrhněte a vysvětlete, jakým způsobem byste v takové situaci nové algoritmy do programu, resp. do uvedené datové struktury, doprogramovávali, případně uveďte nějaké vhodné rozšíření dané datové struktury, které vám přidávání takových algoritmů usnadní (pokud budete potřebovat nějaké další datové typy, tak je popište též).
[1 bod]

Otázka č. 2

Předpokládejte následující program, který pro libovolně velký textový dokument předaný jako 1. argument na příkazovém řádku vypíše počet výskytů slova předaného jako 2. argument:

```
class Prg2 {
    static void Main(string[] args) {
        string inputContent = File.ReadAllText(
            args[0]
        );
        var words = inputContent.Split(
            ' ', '\t', '\n'
        );
        inputContent = null;
        CountWord(words, args[1]);
    }

    static void CountWord(
        string[] words, string word) {
        ...
    }
}
```

(A) Je uvedená část programu pro danou situaci vhodně navržená? Detailně okomentujte a vysvětlete případná pozitiva i negativa řešení.
[1 bod]

(B) Pokud program vypadá uvedeným způsobem, je v něm důležitý řádek `inputContent = null`? Změnilo by se nějak chování programu z libovolného úhlu pohledu, pokud bychom z programu tento řádek vypustili? Detailně vysvětlete proč a případně jak.
[1 bod]

(C) V programu použitá metoda `Split` je metodou s proměnným počtem parametrů. Napište, jak nejspíš vypadá její deklarace (hlavička) v jazyce C# – tj. včetně návratové hodnoty a popisu všech parametrů.
[1 bod]

Otázka č. 3

Je možné v C# zapsat následující deklaraci lokální proměnné?

```
bool b = new System.Boolean();
```

Pokud ano, tak vysvětlete, co daný zápis znamená. Pokud ne, tak vysvětlete proč.
[1 bod]

Otázka č. 4

V abstraktní třídě `Stream` (předek např. třídy `FileStream`) je deklarována následujícím způsobem vlastnost `Length` vracející délku streamu v bytech:

```
public abstract long Length { get; }
```

(A) Je toto v kontextu třídy `FileStream` dle dnešních měřítek vhodné využití C#/CLR konceptu vlastnosti/property, nebo by bylo lepší raději zvolit implementaci běžnou metodou (např. viz níže)?

```
public abstract long GetLength();
```

Detailně vysvětlete proč.
[1 bod]

(B) Je u této vlastnosti volba typu `long` vhodná? Nebylo by lepší použít typ `ulong`, když délka streamu nemůže být nikdy záporná? Vysvětlete proč.
[1 bod]

Otázka č. 5

Kolik jedniček vypíše následující program na standardní výstup? Detailně vysvětlete proč.

```
class Prg5 {
    class X {
        public X() {
            Console.Write("1");
        }
    }

    public static void Main() {
        for (int i = 1; i <= 3; i++) {
            X[] a = { new X(), new X() };
        }
    }
}
```

[1 bod]

Otázka č. 6

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

```
class A {
}

class B {
    public static implicit operator A(B b) {
        return new C();
    }
}

class C : A {
    public static implicit operator B(C c) {
        return new D();
    }
}

class D : B {
}

class Prg6 {
    private static void m(A a) {
        Console.WriteLine("6.1");
    }

    private static void m(C c) {
        Console.WriteLine("6.2");
    }

    private static void m(D d) {
        Console.WriteLine("6.3");
    }

    public static void Main() {
        m(new B());
    }
}
```

[1 bod]