Následující program byl spuštěn bez parametrů příkazové řádky a zastavil se na vyznačené řádce, která ještě nebyla vykonána. Zapište celý obsah paměťové oblasti zásobník a halda. U všech pojmenovaných entit uveďte **datový typ**, **jméno** a **hodnotu**, u nepojmenovaných zapište třídu a všechny atributy (**typy**, **názvy** a **hodnoty**). Pokud je hodnotou reference, pak zakreslete **šipku** k instanci, na kterou reference odkazuje. Vyznačte **zásobníkové rámce**.

```
class Green {
  Yellow next;
  public void Attach(int n) {
    if (n>0) {
      this.next = new Yellow();
      this.next.Attach(n-1);
    }
    else
      Console.WriteLine("Mamba is complete!");
  }
}

class Yellow {
  Green next;
  public void Attach(int n) {
    if (n>0) {
      this.next = new Green();
      this.next.Attach(n-1);
    }
    else
      Console.WriteLine("Mamba is complete!");
  }
}

public class Snake {
  Green first;
  public Snake(int n) {
    this.first = new Green();
    this.first.Attach(n-1);
  }

  public static void Main(String[] args) {
    Snake s = new Snake(3);
  }
}
```

Je dán následující program:

```
class ClassA {

  public static void Main(String[] args) {
    ClassB b = new ClassB();
    printWhoIsIt(b);
  }

  static void printWhoIsIt(IWriter p) {
    Console.WriteLine(p.whoAreYou());
  }
}
```

Doplňte zdrojový kód tak, aby výsledkem volání uvedené metody `Main` byl následující výpis:

```
I am ClassB
```

Program nesmí skončit předčasně!

Následující program byl spuštěn bez parametrů příkazové řádky a zastavil se na vyznačené řádce, která ještě nebyla vykonána. Zapište celý obsah paměťové oblasti zásobník a halda. U všech pojmenovaných entit uveďte **datový typ**, **jméno** a **hodnotu**, u nepojmenovaných zapište třídu a všechny atributy (**typy**, **názvy** a **hodnoty**). Pokud je hodnotou reference, pak zakreslete **šipku** k instanci, na kterou reference odkazuje. Vyznačte **zásobníkové rámce**.

```
public class Anteater {
  int[] ants;
  public int[] antsEaten;

  public Anteater(int[] ants) {
    this.ants = ants;
    antsEaten = new int[ants.length];
    this.EatAnts(0);
  }

  public void EatAnts(int s){
    int ant = ants[s];
    if(ant >= ants.length) {
      Console.WriteLine("I am full, yum-yum.");
      return;
    }
    antsEaten[s] = 1;
    EatAnts(s+1);
  }

  public static void Main(String[] args) {
    int[] ants = new int[]{1, 2, 3, 4};
    Anteater a = new Anteater(ants);
    for (int i = 0;i<a.antsEaten.Length;i++)
      Console.WriteLine(a.antsEaten[i]);
  }
}
```

Mějme následující program:

```
interface IMyInterface {
  public void PrintSomething();
}

class MyClass : IMyInterface{
  public void PrintSomething() {
    Console.WriteLine(”42”);
  }
  public void PrintSomethingElse() {
    Console.WriteLine(”24”);
  }
}
```

U každé z následujících řádek určete, zda je v pořádku, popř. zda způsobí chybu při překladu nebo chybu za běhu programu. Při uvažování každé další řádky předpokládejte, že bezchybné předchozí řádky byly vykonány, zatímco chybové řádky v programu vůbec nebyly.

	OK	Chyba při překladu	Chyba za běhu
`MyInterface mi1 = new MyInterface();`			
`MyInterface mi2 = new MyClass();`			
`MyClass mc1 = new MyClass();`			
`MyClass mc2 = (MyClass)mi2;`			
`mi2.PrintSomething();`			
`mi2.PrintSomethingElse();`			
`mc2.PrintSomething();`			
`mc2.PrintSomethingElse();`			

(v každé řádce tabulky zapište jeden křížek do sloupce odpovídajícího situaci, která nastane. Uvažujte program vykonaný po řádcích kromě těch, které by skončily chybou)

Zapište metodu void `sort(IComparable[] data)`, která provede seřazení předaného pole dat libovolným řadicím algoritmem. Rozhraní `IComparable` poskytuje metodu `int CompareTo(Object o)`, která vrací záporné číslo když instance nad kterou je volána patří před instanci `o` a kladné číslo, pokud patří za ni. Nepoužívejte standardní metodu `Array.Sort()`.

Následující třída reprezentuje záznam o osobě a počtu bodů získaných ze tří úloh:

```
class Person ____________________________________
{
  String name; // jmeno
  int points1; // pocet bodu z ulohy 1
  int points2; // pocet bodu z ulohy 2
  int points3; // pocet bodu z ulohy 3









}
```

Reference na instance této třídy jsou uloženy v poli nazvaném `students`, a máme v úmyslu je seřadit následujícím voláním:

```
Array.Sort(students);
```

Doplňte do třídy `Person` co je nutné pro to, aby tímto voláním došlo k seřazení studentů v poli primárně podle celkového počtu bodů, a v případě shody podle abecedy.