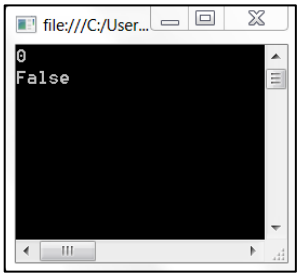
# КОНСТРУКТОРИ

**Основне карактеристике:**

- Конструктор је специјална метода – метод унутар класе са посебним карактеристикама .
- Конструктор се имплицитно позива сваки пут када користимо кључну реч **new** да би креирали инстанцу класе тј. објекат.
- Класа може да има један или више конструктора, који се међусобно разликују по броју параметара.
- Класа не мора да садржи конструктор.
- Ако класа не садржи конструктор, C# (компајлер) аутоматски генерише конструктор, који називамо подразумевани конструктор, default конструктор или конструктор без параметара. То је метод без параметра и са празним телом.
- Конструктори служе за иницијализацију објекта (постављање вредности атрибута) и заузимање меморијски простор (алокацију меморијског простора) за објекат.
- Име конструктора је исто као и име класе.
- Конструктори не враћају вредност, без кључне речи void.
- Подразумевани конструктор иницијализује сва поља класе: **нумеричке вредности** на 0, bool (**логичке променљиве**) на false, char na '\0', **референце** на null.
- Ако постоји бар један конструктор у класи, подразумевани конструктор се не генерише.
- Код изведених класа, када се креира објекат изведене класе прво се позива основна или родитељска класа.

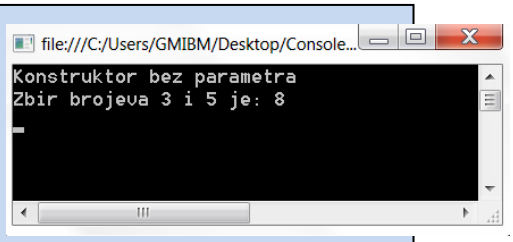
**Пример**: Илустровати пример када се имплицитно позива конструктор – конструктор без параметара. Креирати класу **A**, дефинисати атрибуте (int и bool). Креирати објекат и приказати вредности атрибута.

```
using System;
    class Program
    {
        static void Main(string[] args)
        {
            A obj = new A();
            Console.WriteLine(obj.a);
            Console.WriteLine(obj.b);
            Console.ReadKey();
        }
    }
    class A {
        public int a;
        public bool b;
    }
```




**Пример:** Илустровати пример када се експлицитно позива конструктор – конструктор без параметара. Такође, дефинисати конструктор са два параметра.

```
using System;
    class Program
    {
        static void Main(string[] args)
        {
            A obj1;
            A obj2 = new A();
            A obj3 = new A(3, 5);
            Console.WriteLine(obj1 == null);  //zasto kompajler prijavljuje gresku?
            Console.ReadKey();
        }
    }
    class A {
        public A() {
            Console.WriteLine("Konstruktor bez parametra");
        }
        public A(int a, int b) {
            Console.WriteLine("Zbir brojeva {0} i {1} je: {2}", a, b, a + b);
        }
    }
```




**Пример:** Дате су две класе, основна и изведена класа. Анализирати код и проценити шта ће бити приказано на екрану?

```
using System;
    class Program
    {
        static void Main(string[] args)
        {
            Izvedena obj = new Izvedena();
            Console.ReadKey();
        }
    }
    class Osnovna {
        public Osnovna()
        {
            Console.WriteLine("Konstruktor bez parametra osnovne klase");
        }
    }
    class Izvedena : Osnovna {
        public Izvedena() {
            Console.WriteLine("Konstruktor bez parametra izvedene klase");
        }
    }
```

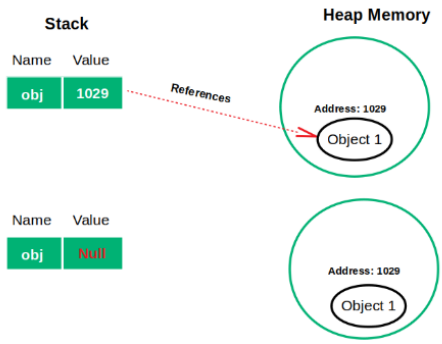
## ДЕСТРУКТОРИ

Конструктори се позивају сваки пут када се креира неки објекат. При уништавању објекта позивају се деструктори. Деструктори су методе који имају само име као и име класе тј. име конструктора са додатком знака ~ (тилда) као првог симбола имена. За разлику од деструктора немају дефинисан модификатор (квалификатор) приступа public. Такође, немају повратну вредност као ми наредбу return у телу. Деструктори немају параметре.

```
~ime_klase
{
        //programski kod
}
```

**НАПОМЕНА:** Кључна реч **null**

Референтом типу се може доделити литерал **null**, што значи да та референца не упућује ни на један објекат.




**Пример:** Креирати класу A, са атрибутом типа int, конструктором без параметра и деструктором. Анализирати програмски код и објаснити шта ће се приказати на екрану? Ако пријављује грешку објаснити зашто се то дешава? Ако изузмемо грешке, који је приказ на екрану?

```
using System;
class Program
{
    static void Main(string[] args)
    {
        A a = new A();
        Console.WriteLine(a == null);
        Console.WriteLine(a.x);
        Console.ReadKey();

        a = null;
        Console.WriteLine(a == null);
        Console.WriteLine(a.x);
        Console.ReadKey();
    }
    class A {
        public int x;

        public A() {
            Console.WriteLine("Konstruktor");
        }
        ~A() {
            Console.WriteLine("Destruktor");
        }
    }
}
```

**НАПОМЕНА:** Ако програмски код изменимо, модификатор приступа **public** заменимо са **private** за атрибут и конструктор, шта ће се десити?

**АНАЛИЗИРАТИ И ДАТИ ТАЧНЕ ОДГОВОРЕ ЗА НАВЕДЕНА ПИТАЊА. ЗА СВАКИ ОДГОВОР ДАТИ ОБЈАШЊЕЊЕ!**

214. Који искази у програмском језику C# дефинишу конструктор: **2**

1. Подразумевани конструктор без параметара се увек аутоматски додаје класи.
2. Подразумевани конструктор без параметара се класи аутоматски додаје уколико у њој није експлицитно дефинисан ниједан конструктор.
3. У класи се мора експлицитно дефинисати бар један конструктор.
4. Конструктори немају тип резултата, чак ни **void.**

155. Дати код програма у програмском језику C# састоји се од две класе у једној датотеци. Анализирати дати код и проценити његову тачност. **1**

```
namespace TestPrimer {
classProgram {
staticvoid Main(string[] args) {
Klasa a = newKlasa();
            a.n++;
        }
    }
classKlasa {
privateint n;
private Klasa() { }
    }
}
```

1. Програм има грешку јер класа ***Klasa*** има приватни конструктор и приватно поље ***n***.
2. Програм има грешку јер класа ***Klasa*** има празан подразумевани конструктор.
3. Програм има грешку јер променљива ***n*** није иницијализована.
4. Програм нема грешака и нормално се извршава

205. Дат је код програма у програмском језику C# у ком су дефинисане три класе: classProgram која садржи Main(string[] args) методу, classA и classB. Анализирати дати код и одредити шта ће се приказати на екрану као резултат извршавања овог програма. Заокружити број испред одговора који садржи резултат исписа: **2**

```
classProgram : A {
publicstaticvoid Main(string[] args) {
Program p = newProgram();
   }
}
classA : B {
public A() { Console.WriteLine("Pozvan podrazumevani konstruktor klase
A"); }
}
classB {
public B() { Console.WriteLine("Pozvan podrazumevani konstruktor klase
B"); }
}
```

1. Ништа.
2. Позван подразумевани конструктор класе A
3. Позван подразумевани конструктор класе B
4. Позван подразумевани конструктор класе A и у другом реду: Позван подразумевани конструктор класе B
5. Позван подразумевани конструктор класе B и у другом реду: Позван подразумевани конструктор класе A

179. Дат је код програма у програмском језику C#. Анализирати дати код и проценити његову тачност:

```
namespace TestPrimer{
classTest {
int x;
publicTest(string s){
Console.WriteLine("Klasa Test");
        }
staticvoid Main(string[] args){
Testt = null;
Console.WriteLine(t.x);
        }
    }
}
```

1. Програм има грешку, јер променљива x није иницијализована.
2. Програм има грешку, јер класа Test нема подразумевани конструктор.
3. Програм има грешку, јер се у некој класи не може декларисати променљива типа те исте класе, као што је то овде случај са променљивом t.
4. Програм има грешку, јер променљива t није иницијализована и има вредност null у моменту када се приказује поље t.x .
5. Програм нема грешака и нормално се извршава, не приказујући ништа на екрану.

**2**

176. При креирању објеката изведене класе:

1. извршава се само конструктор изведене класе
2. прво се извршава конструктор родитељске класе, али само ако је позван кључном речју ***base***
3. обавезно се прво извршава конструктор изведене, а потом конструктор родитељске класе
4. обавезно се прво извршава конструктор родитељске, а потом конструктор изведене класе

**1**

156. У програмском језику C# декларисане су статичке променљиве логичког, нумеричког и класног типа.

```
namespace Test{
classProgram
    {
publicstaticbool x;
publicstaticint y;
publicstaticRandom r;
staticvoid Main(string[] args) { … }
    }
}
```

Одредити које аутоматске почетне (default) вредности декларисана поља имају унутар методе Main. Понуђени одговори су наведени у наглашеном редоследу:

| | x | y | r |
|---|---|---|---|
| 1. | true | 1 | null |
| 2. | false | 0 | null |
| 3. | true | 0 | null |
| 4. | false | 1 | null |
| 5. | false | 0 | void |

**1**

181. Дат је код програма у програмском језику C#:

```
namespace TestPrimer{
classProgram
    {
staticvoid Main(string[] args)
        {
Console.WriteLine(fun(17));
        }
publicint fun(int n) { return n; }
publicvoid fun(int n){ Console.WriteLine(n); }
    }
}
```

Анализирати код и заокружити број испред очекиваног одговора:

1. Програм има грешку, јер се не може одредити коју верзију преоптерећеног метода **fun(...)** треба позвати.
2. Програм има грешку, јер је друга верзија преоптерећеног метода **fun(...)** дефинисана, али се нигде не позива.
3. Програм се нормално извршава и приказује 17 једанпут.
4. Програм се нормално извршава и приказује 17 двапут.

**2**

210. У класи **Figura** дат је подразумевани (default) конструктор и конструктор са 4 параметра:

```
public Figura() {...}
public Figura(string ime, string boja, int pozX, int pozY) {...}
```

Заокружити бројеве испред исправно написаних наредби креирања објеката класе Figura:

1. `Figura f = Figura("lovac", "beli", 7, 3);`
2. `Figura f = newFigura("beli", "lovac", 7, 3);`
3. `Figura f = newFigura();`
4. `Figura f = newFigura("lovac", 3, 7, "beli");`
5. `Figura f = newFigura("lovac", "beli", 3, 7);`
6. `Figura f = newFigura("lovac", "beli", 3);`

**1,5**

236. Са леве стране су наведени делови/елементи класе, а са десне стране улоге појединих класних елемената. На линију испред описа улоге унети редни број под којим је наведен одговарајући елеменат класе:

| | | |
|---|---|---|
| 1. поље (атрибут) | _____ | Опис функционалности објеката класе |
| 2. деструктор | _____ | Контрола приступа пољима класе |
| 3. конструктор | _____ | Опис особина објеката класе |
| 4. метод | _____ | Креирање објеката класе |
| 5. својство / property | _____ | Уништавање објеката класе |

**2,5**