Platforma .NET

Laboratorium nr 1 – Podstawy języka C#

Ćwiczenie 1

1. Utwórz nowy projekt
a. Z menu **File** wybierz **New/Project...**

b. W oknie dialogowym **New Project** określ następujące właściwości**:**

- typu projektu: **Visual C#/Windows**
- szablon: **Console Application**
- lokalizacja: **D:\nr_grupy_nazwisko\visual studio projects**
- nazwa projektu: **Cwiczenie_1.**
- nazwa rozwiązania: **Lab1**

2 Do metody Main dodaj następujący kod.

```
Console.WriteLine("Ahoj Przygodo !!!");
Console.ReadKey(true);
Console.Write("Koniec");
Console.ReadKey();
```

3. Skompiluj i uruchom program.
* Czym różnią się wywołania funkcji ReadKey? (F1 – interaktywna pomoc VS)

Ćwiczenie 2

1. Spróbuj skompilować poniższy program.

2. Korzystając z okna **Error List** lub **Output** popraw błędy w programie. Po poprawieniu błędów program może wyglądać następująco:

```
namespace Bledy
{
    class Program
    {
        static void main(string[] args)
        {
            Console.WriteLine("Zrobił wilk elektrownię, lecz by prąd uzyskać);
            Console.WriteLine("Spalał w niej cały węgiel z kopalni od liska.")
            Console.WriteLine(Kopalnia z elektrowni cały prąd zżerała,");
            Console.WriteLine("Stąd brak światła i węgla. Ale system działa!");
            Console.WriteLine();
            Console.WriteLine("Andrzej Waligórski")
            ReadLine();
        }
}
```

Ćwiczenie 3

1. Wewnątrz metody Main napisz następujący kod:
a. Zadeklaruj trzy zmienne rzeczywiste a, b, tmp. Zmienna a i b są to te zmienne, między którymi będziemy wymieniać wartości, zmienna tmp jest zmienną pomocniczą:

```
double a, b, tmp;
```

b. Pobierz od użytkownika wartości zmiennych a i b:

```
Console.Write("Podaj wartość zmiennej a: ");
a = Convert.ToDouble(Console.ReadLine());
Console.Write("Podaj wartość zmiennej b: ");
b = Convert.ToDouble(Console.ReadLine());
```

c. Wymień wartości zmiennych `a` i `b`, przy pomocy zmiennej `tmp`.

```
tmp = a; a = b;
b = tmp;
```

`d.` Wypisz wartości zmiennych po wymianie, a następnie zatrzymaj działanie programu:

```
Console.WriteLine("a = {0}, b = {1}", a, b);
Console.ReadKey();
```

2. Skompiluj i uruchom program.

Ćwiczenie 4

Program obliczający cenę brutto oraz podatek od podanej ceny netto..

1. Wewnątrz metody Main napisz następujący kod:

`a.` Zadeklaruj stałą symboliczną rzeczywistą o dużej precyzji o nazwie `Vat` i nadaj jej wartość 0,22.

```
const decimal Vat = 0.22M;
```

b. Zadeklaruj zmienną rzeczywistą o dużej precyzji `netto`.

```
decimal netto;
```

`c.` Pobierz od użytkownika cenę netto:

```
Console.Write("Podaj cenę netto: ");
netto = Convert.ToDecimal(Console.ReadLine());
```

`d.` Zadeklaruj zmienne `podatek` oraz `brutto` i odpowiednio je zainicjalizuj..

```
decimal podatek = Vat * netto;
decimal brutto = netto + podatek;
```

`e.` Wypisz obliczone wartości podatku i ceny brutto, dodając symbol waluty, a następnie zatrzymaj działanie programu:

```
Console.WriteLine("Cena wynosi {0:C}, w tym ¬kwota podatku: {1:C}.", brutto, podatek);
Console.ReadKey();
```

2. Skompiluj i uruchom program.

Ćwiczenie 5

Program obliczający pole i obwód koła o zadanym promieniu

1. Wewnątrz metody Main napisz następujący kod:
`a.` Zadeklaruj zmienną rzeczywistą r.

```
double r;
```

b. Pobierz od użytkownika długość promienia:

```
Console.Write("Podaj długość promienia: ");
r = Convert.ToDouble(Console.ReadLine());
```

c. Zadeklaruj zmienne `pole` oraz `obwod` i odpowiednio je zainicjalizuj. We wzorach na obwód i pole skorzystaj ze stałej `Math.PI`.

```
double pole = Math.PI * r * r;
double obwod = 2 * Math.PI * r;
```

d. Wypisz obliczone wartości pola i obwodu koła z dokładnością do trzech miejsc po przecinku, a następnie zatrzymaj działanie programu:

```
Console.WriteLine("Pole koła o promieniu {0} ¬wynosi: {1:f3}.", r, pole);
Console.WriteLine("Obwód koła o promieniu {0} ¬wynosi: {1:f3}.", r, obwod);
Console.ReadKey();
```

2. Skompiluj i uruchom program.

Ćwiczenie 6

Napisz program, który oblicza wartość wyrażenia:

$$z = \begin{cases} xy & dla \quad x < 0 \; i \; y < 0 \\ 10 & dla \quad x = 0 \text{ lub } y = 0 \\ x + y & dla \quad pozostalych \quad wartosci \end{cases}$$

Wewnątrz metody Main napisz następujący kod:

a. Zadeklaruj cztery zmienne rzeczywiste x, y, z, v.

```
double x,y,z,v;
```

b. Pobierz od użytkownika wartości zmiennych x, y:

```
Console.Write("Podaj wartość zmiennej x: ");
x = Convert.ToDouble(Console.ReadLine());
Console.Write("Podaj wartość zmiennej y: ");
y = Convert.ToDouble(Console.ReadLine());
```

c. Sprawdź czy zmienna x i zmienna y nie są mniejsze od zera. Jeżeli są, do zmiennej z wstaw wartość iloczyny xy.

```
if(x < 0 && y < 0) { z = x * y; }
```

d. W przeciwnym wypadku sprawdź czy zmienna x lub zmienna y ma wartość zero. Gdy tak jest, do zmiennej z wstaw wartość 10, natomiast w przeciwnym razie do zmiennej z wstaw sumę x + y.

```
else { if(x == 0 || y == 0)
            {
                  z = 10;
            }
            else { z = x + y; }
}
```

e. Wypisz wartość zmiennej z, a następnie zatrzymaj program, aby użytkownik mógł obejrzeć wyniki.

```
Console.Write(" z = {0}" , z);
Console.ReadKey();
```

2. Skompiluj i uruchom program.

Ćwiczenie 7
Program wyznaczający maksimum z dwóch liczb całkowitych przy pomocy wyrażenia warunkowego.

3. Wewnątrz metody Main napisz następujący kod:

a. Zadeklaruj zmienne całkowite: a, b, max.

```
int a, b, max;
```

b. Pobierz od użytkownika wartości zmiennych a i b:

```
Console.Write("Podaj pierwszą wartość: ");
a = Convert.ToInt32(Console.ReadLine());
Console.Write("Podaj drugą wartość: ");
b = Convert.ToInt32(Console.ReadLine());
```

c. Wyznacz wartość maksymalną z liczb a i b:

```
max = a > b ? a : b;
```

d. Wypisz obliczoną wartość maksymalną, a następnie zatrzymaj działanie programu:

```
Console.WriteLine("Wartość maksymalna wynosi: ¬{0}.", max);
Console.ReadKey();
```

4. Skompiluj i uruchom program

Ćwiczenie 8
Utwórz programu, który obliczy pierwiastki równania kwadratowego: $ax^2+bx+c=0$.

3. Wewnątrz metody Main napisz następujący kod:
a. Zadeklaruj następujące zmienne rzeczywiste a, b, c, x1, x2, delta. Zmienna a, b, są parametrami równania kwadratowego, w zmiennych x1, x2 będziemy przechowywać wartości pierwiastków, zmienna delta jest zmienną pomocniczą:

```
double a, b, c, x1, x2, delta;
```

b. Ustaw blok try catch. W bloku catch przechwyć wszystkie wyjątki i wypisz wiadomość związaną z zgłoszonym wyjątkiem:

```
try {
      ...
}
catch(Exception ex) {
      Console.WriteLine("Program został przerwany. ¬{0}", ex.Message);
}
```

c. Po bloku catch zatrzymaj program, aby użytkownik mógł obejrzeć wyniki.

```
...
Console.ReadKey();
```

Dalsza część kodu będzie umieszczona w bloku try
d. Pobierz od użytkownika wartość parametru a. Jeżeli wartość jest równa zero zgłoś wyjątek.

```
...
Console.Write("Podaj wartość parametru a: ");
a = Convert.ToDouble(Console.ReadLine());
if(a == 0) {
      throw new Exception("Parametr a powinien być ¬różny od zera");
}
```

e. Pobierz od użytkownika wartości parametrów b i c:

```
...
Console.Write("Podaj wartość parametru b: ");
b = Convert.ToDouble(Console.ReadLine());
Console.Write("Podaj wartość parametru c: ");
c = Convert.ToDouble(Console.ReadLine());
```

f. Oblicz wartość parametru delta dla podanych parametrów.

```
delta = b * b - 4* a * c;
```

g. Jeżeli delta jest większa od zera oblicz oba pierwiastki i wypisz ich wartości na ekranie.

```
...
if(delta > 0) {
      x1 = (-b - Math.Sqrt(delta))/(2*a);
      x2 = (-b + Math.Sqrt(delta))/(2*a);
      Console.WriteLine("Równanie ma dwa pierwiastki:");
      Console.WriteLine("\tx1 = {0}", x1);
      Console.WriteLine("\tx2 = {0}", x2);
}
```

h. W przeciwnym razie sprawdź czy delta jest równa 0 i oblicz pojedynczy pierwiastek. Jeżeli nie, wypisz, że równanie nie ma pierwiastków rzeczywistych

```
...
else {
      if(delta == 0) {
            x1 = -b/(2*a);
            Console.WriteLine("Równanie ma jeden pierwiastek rzeczywisty:");
      Console.WriteLine("\tx1 = {0}", x1);
      }
      else {
            Console.WriteLine("Równanie nie ma pierwiastków rzeczywistych.");
      }
}
```

4. Skompiluj i uruchom program.

Ćwiczenie 9

Napisz program obliczający silnię z n liczb. N ma zostać podane jako parametr funkcji `Silnia()`.

1. Wewnątrz metody Main napisz następujący kod:
a. Zadeklaruj zmienne całkowite nieujemne n oraz silnia:

```
ulong silnia;
ushort n;
```

b. Pobierz od użytkownika wartości zmiennej n:

```
Console.Write("Podaj wartość zmiennej n: ");
n = Convert.ToUInt16(Console.ReadLine());
```

c. Poza metodą Main stwórz statyczną metodę Silnia() z deklaracją zmiennej lokalnej wart_silnia. Wartość ta zgodnie z zasadą działania funkcji silnia, zainicjalizowana jest wartością 1.

```
static void Main(string[] args){
      ...
}
static ulong silnia(ushort n) {
```

```
      ulong wart_silnia=1;
      ...
}
```

d. Korzystając z pętli for(){} oblicz wartość silni dla liczby n.
i. Zapewnij, że w razie przepełnienia, zostanie zgłoszony wyjątek.

```
for(ushort i = 1; i <= n; i++) {
      checked {
            wart_silnia=wart_silnia*i;
      }
}
```

e. Zwróć wartość obliczonej silni z funkcji do głównej metody Main:

```
return wart_silnia;
```

f. Wywołaj w głównej metodzie Main funkcję silnia dla podanej liczby n (* w jakie inne sposoby można wywołać statyczną funkcję Silnia()?):

```
silnia=Silnia(n);
```

g. Wypisz wartości zmiennej silnia, a następnie zatrzymaj program, aby użytkownik mógł obejrzeć wyniki.

```
Console.Write("Wartość silni z liczby "+n+" to "+ silnia);
Console.ReadKey();
```

2. Skompiluj i uruchom program.
3. Wykonaj powyższe zadanie z użyciem pętli `while(){}` oraz `do{} while();`
* Wykonaj powyższe zadanie z użyciem rekurencji oraz instrukcji switch:

```
      switch(n){
      case 0:
            wart_silnia=1;
            break;
      case 1:
            wart_silnia=1;
            break;
      default:
            return n*silnia((ushort)(n-1));
      }
```

Ćwiczenie 10

Napisz program, grę - zgadnij liczbę. Użytkownik ma zgadnąć liczbę wylosowaną przez komputer. Do wylosowania liczby pseudolosowej stosuj następujący kod.

```
Random generator = new Random();
int liczba = generator.Next(a, b +1);
```

Do zmiennej liczba zostanie podstawiona wylosowana wartość całkowita z przedziału `<a; b>`.
1. Dodaj do bieżącego rozwiązania nowy projekt
2. Uczyń nowo utworzony projekt projektem startowym
a. Zaznacz projekt **Gra** w okienku **Solution Explorer** i z menu kontekstowego wybierz **Set as StartUp Project.**

3. Wewnątrz metody Main napisz następujący kod:
`a.` Zdefiniuj początek i koniec przedziału z którego będą generowane liczby.

```
const int a = 0;
const int b = 200;
```

b. Zadeklaruj następujące zmienne całkowite: n i m. n - liczba wygenerowana przez generator liczb pseudolosowych; m - liczba podana przez użytkownika

```
int n, m;
```

c. Wylosuj wartość zmiennej n.

```
Random generator = new Random();
n = generator.Next(a, b +1);
```

d. Wykonuj następujące instrukcje, póki użytkownik nie zgadnie wylosowanej wartości:
i. Pobierz wartość zmiennej m od użytkownika.
ii. Poinformuj użytkownika, czy podana przez niego liczba jest za duża albo za mała.

```
do
{
      Console.WriteLine("Podaj wartość wylosowanej liczby. Wartość jest z
      przedziału <{0}, {1}>: ", a, b);
      m = Convert.ToInt32(Console.ReadLine());
      if(m < n)
            Console.WriteLine("Wartość podana jest za mała");
      if(m > n)
            Console.WriteLine("Wartość podana jest za duża");
}
while(m != n);
```

e. Pogratuluj użytkownikowi zgadnięcia wylosowanej liczby. Zatrzymaj program, aby użytkownik mógł obejrzeć wyniki.

```
Console.WriteLine("Gratulacje!!! Odgadłeś liczbę.");
Console.ReadKey();
```

4. Skompiluj i uruchom program.