**Оператори за вход и изход:**

За въвеждане на данни – Console.ReadLine();

За извеждане на данни на един ред –> Console.Write(<стойности>);

За извеждане на данни с преминаване на нов ред –> Console.WriteLine(<стойности>);

За извеждане на данни с повече от една стойност –> Console.WriteLine(“Age = {0}; City = {1}”, age, city);

**Математически оператори:**

\* –> умножение

/ –> деление

% –> деление с остатък

= –> присвояване (var a = 5; => на променливата а се присвоява стойността 5)

>= ­–> дали изразът отляво е по-голям или равен на израза отдясно ( 5 >= 2 \* a) => False

== –> проверка за равенство

!= –> проверка за неравенство

**Типове Данни:**

int – цяло число

double – дробно/десетично число

от текст към цяло число => int <име на променлива> = int.Parse(<стойност от тип текст>);

П-р: int a = int.Parse(“50”);

int b = int.Parse(Console.ReadLine());

Аналогично за дробни числа използваме double.Parse(<стойност от тип текст>);

Оператори за проверка:

**Оператор If:**

if ( <условие> )

{

<операторите се изпълняват, ако условието е истина>;

}

П-р: var grade = 5.50;

if (grade >= 5.50)

{

Console.WriteLine(“Excellent!”);

}

**Оператор if-else**

if ( <условие> )

{

<операторите се изпълняват, ако условието е истина>;

}

else

{

<операторите се изпълняват, ако условието е неистина/лъжа/грешно>;

}

**Поредица от if-else:**

if (<условие>)

{

<операторите се изпълняват, ако условието е истина>;

}

else if (<условие2>)

{

<операторите се изпълняват, ако условие2 е истина>;

}

else if (<условие3>)

{

<операторите се изпълняват, ако условие3 е истина>;

}

else

{

<операторите се изпълняват във всички останали случаи>;

}

Можете да изписвате колкото искате пъти else-if, като накрая не сте длъжни да има финален else, който да покрива останалите случаи.

П-р: Задача 6 от документ 06.Серия-от-проверки-упражнения:

if (speed <= 10)

{

Console.WriteLine("slow");

}

else if (speed > 10 && speed<=50)

{

Console.WriteLine("average");

}

else if (speed > 50 && speed <=150)

{

Console.WriteLine("fast");

}

else if (speed > 150 && speed <= 1000)

{

Console.WriteLine("ultra fast");

}

else

{

Console.WriteLine("extremely fast");

}

**Вложени оператори if-else:**

if (<условие1>)

{

<операторите се изпълняват, ако условие1 е истина>;

if (<условие2>)

{

<операторите се изпълняват, ако условие1 и условие2 са истина>;

}

else

{

<операторите се изпълняват, ако условие1 е истина, но условие 2 НЕ е вярно>;

}

}

Оператор за проверка switch:

switch (<променлива със стойност>)

{

case <стойност 1> : <оператори, които ще се изпълнят за стойност1>; break;

case <стойност 2> : <оператори, които ще се изпълнят за стойност2>; break;

…….

default: <оператори, които се изпълняват във всички останали случаи, т.е. стойността е различна от зададените в case стойности>; break;

}

П-р: Въведени са две числа a и b, и символ за математическа операция между тях (‘+’, ‘–’, ‘\*’, ‘/’). Да се извърши действието и отпечата резултата.

int a = 6, b = 10;

char symbol = char.Parse(Console.ReadLine());

switch(symbol)

{

case ‘+’: Console.WriteLine(a + b); break;

case ‘–’: Console.WriteLine(a – b); break;

case ‘\*’: Console.WriteLine(a \* b); break;

case ‘/’: Console.WriteLine(a / b); break;

default: Console.WriteLine(“Our calculator supports only these operations: ‘+’, ‘–’, ‘\*’, ‘/’ ”); break;

}

П-р: Да се въведе число от 0 до 10 и да се изведе съобщение дали е четно или нечетно.

int number = int.Parse(Console.ReadLine());

switch (number)

{

case 0:

case 2:

case 4:

case 6:

case 8:

case 10: Console.WriteLine("chetno"); break;

case 1:

case 3:

case 5:

case 7:

case 9: Console.WriteLine("nechetno"); break;

default:

Console.WriteLine("Ne si vyvel chislo m/u 0 i 10");

break;

}

Оператори за цикъл:

1. for (<инициализация>; <условие за край>; <актуализация>)

{

<тяло на цикъла => командите, които ще се повтарят>

}

Първо се извършва инициализацията, при която задаваме стойност на управляваща променлива. След това се проверява условието за край и докато то е вярно, се изпълнява тялото на цикъла. При всяко завършване на тялото се изпълнява актуализацията, при която променяме стойността на управляващата променлива. След което се проверява отново условието за край.

Задача: Да се отпечатат числата от 1 до 100 на екрана.

for (int i = 1; i <= 100; i++)

{

Console.WriteLine(i);

}

**Вложени for цикли**

for (<инициализация1>; <условие за край1>; <актуализация1>)

{

for (<инициализация2>; <условие за край2>; <актуализация2>)

{

<тяло на цикъла2 => командите, които ще се повтарят>

}

// може да има още оператори в тялото на първия for цикъл

}

Първо се извършва инициализацията на първия/външния for цикъл, при която задаваме стойност на управляваща променлива. След това се проверява условието за край на външния цикъл и докато то е вярно, се изпълнява тялото на цикъла, в което имаме вложен for цикъл. Извършва инициализацията на втория/вложения for цикъл, при която задаваме стойност на управляващата му променлива. След това се проверява условието за край на вложения цикъл и докато то е вярно, се изпълнява тялото на вложения цикъл. Когато се достигне условието за край на вложения for цикъл, се изпълнява актуализацията на външния for цикъл.

1. **Оператор while**

<инициализация>

while(<условие >)

{

<тялото на цикъла>

<актуализация>

}

Задача: Да се отпечатат числата от 1 до 100 на екрана.

**int i = 1;**

**while (i <= 100)**

**{**

**Console.WriteLine(i);**

**i++;**

**}**

По-удобен за използване е for, когато знаем колко пъти ще се повтарят действията в тялото на цикъла! Когато обаче не знаем колко пъти ще се извършват действията – използваме whilе.

Можете да проверявате няколко условия в while и for, използвайки **логическото умножение (и) (&&)** и **логическото събиране (или) (||)**

1. **do … while**

<инициализация>

do

{

<тяло на цикъла>

<актуализация>

} while (<условие>)**;**

Използва се тогава, когато желаем операциите в цикъла да се изпълнят поне веднъж (дори условието да е грешно)!

П-р: Въвеждайте поредица от числа, докато не се въведе числото 0. Отпечатайте въведените числа.

int n = int.Parse(Console.ReadLine());

while (n != 0)

{

Console.WriteLine(“Printed: {0}”, n);

n = int.Parse(Console.ReadLine());

}

int n2 = int.Parse(Console.ReadLine());

do  
{

Console.WriteLine(“Printed: {0}”, n2);

n2 = int.Parse(Console.ReadLine());

} while(n2 != 0);

Полезни проверки:

1. намиране на четно число => if (<променлива> % 2 == 0) { <действия с числото> }
2. намиране на нечетно число => if (<променлива> % 2 != 0) { <действия с числото> }

Пример: int n = int.Parse(Console.ReadLine());

if (n % 2 == 0) { Console.WriteLine(“even”);}

else { Console.WriteLine(“odd”); }

1. намиране на четно положително число =>

if(<променлива> % 2 == 0 && <променлива> > 0) {<действия>}

Пример: if (n % 2 == 0 && n > 0) { Console.WriteLine(“positive even number”);}

1. намиране на нечетно и отрицателно число =>

if(<променлива> % 2 != 0 && <променлива> < 0) {<действия>}

1. броим числата, които са кратни на 3 или 2 =>

int br=0;

if(<променлива> % 3 == 0 || <променлива> % 2 == 0) { br++; }

1. намиране на най-голямо (максимално) число между 2 числа:

int n1 = 10, n2 = 20;

if (n1 > n2) { Console.WriteLine(n1);}

1. намиране на най-голямо и най-малко число измежду поредица от **отрицателни** числа:

int n = int.Parse(Console.ReadLine());

int max = n;

int min = n;

while (**n < 0**)

{

if (n > max) // намерили сме по-голямо

{

max = n;

// записваме стойността на n в променливата max

}

if (n < min)

{

min = n;

}

// Актуализация

n = int.Parse(Console.ReadLine());

}

Console.WriteLine("Max: {0}", max);

Console.WriteLine("Min: {0}", min);

1. действия с цифрите на число

int n = int.Parse(Console.ReadLine());

int cifra;

while (n != 0)

{

c = n % 10;

<действия с цифрата>

n = n / 10;

}

1. ьььь

Алгоритъм за решаване на задачи:

1. Прочитаме условието толкова пъти, колкото е нужно, за да сме сигурни, че го разбираме.
2. Отбелязваме входните ни данни и създаваме променливи за тях.
3. Имаме ли нужда от междинни данни/променливи
4. Алгоритъмът за самата задача

* проверяваме за ограничения на нашите стойности
* извършваме преобразувания

1. Извеждаме резултат(и)
2. Проверяваме с различни данни
3. Използваме операторите/командите, за да извършим действията, които описахме с думи.