giganci programowania

Materiały powtórzeniowe

Lekcja 3. Operacje matematyczne

Witaj w materiałach powtórzeniowych przygotowanych specjalnie dla Ciebie. Znajdziesz tutaj powtórkę materiału z ostatniej lekcji oraz zadania do wykonania dla chętnych. Z pewnością sobie poradzisz. Powodzenia!

Cel lekcji

Celem lekcji jest przedstawienie najczęściej stosowanych operacji matematycznych.

Operacje arytmetyczne

W kodzie najczęściej wykorzystujemy podstawowe działania matematyczne takie jak:

- Dodawanie
- Odejmowanie
- Mnożenie
- Dzielenie
- Modulo – reszta z dzielenia

Nazwa działania	Operator	Opis	Przykład	Wynik
Dodawanie	**+**	Dodanie jednej wartości do drugiej	**12+3**	**15**
Odejmowanie	**-**	Odjęcie jednej wartości od drugiej	**5-2**	**3**
Dzielenie	**/**	Podzielenie dwóch wartości	**10/2**	**5**
Mnożenie	*****	Pomnożenie dwóch wartości	**10*5**	**50**
Modulo	**%**	Podzielenie dwóch wartości i zwrot reszty z dzielenia	**10%2**	**0**

Przykład

```
int a = 5;
int b = 10;
int c = 4;

//dodawanie
int suma = a + b + c;
Console.WriteLine(suma);

//odejmowanie
int roznica = b - c;
Console.WriteLine(roznica);

//mnożenie
int iloczyn = a * b + c;
Console.WriteLine(iloczyn);

//dzielenie
int iloraz = b / a;
Console.WriteLine(iloraz);

//modulo
int modulo = b % c;
Console.WriteLine(modulo);

//kolejność wykonywania działań
int kwd = (a + b * c) / c;
Console.WriteLine(kwd);

Console.ReadLine();
```

Utrata precyzji podczas działań matematycznych.

Istotną rolę podczas pisania programu, w którym występują działania matematyczne, odgrywa dobór odpowiednich typów zmiennych liczbowych.

Powiedzmy, że robimy prosty kalkulator i chcemy wykonać dzielenie:

5 / 2 – wynik jest oczywisty – 2.5, ale co jeśli użyjemy tu zmiennych, które nie przechowują wartości „po przecinku”?

```
int a = 5;
int b = 2;

Console.WriteLine(a / b);
Console.ReadLine();
```

W wyniku tego kodu dostaniemy 2, czyli nasze obliczenia mocno straciły na precyzji.

O ile nasz mały kalkulator co najwyżej sprawi komuś kłopot na zajęciach o tyle w programach obsługujących np. odmierzanie odpowiednich proporcji leków, czy obsługujących reaktor atomowy, taki błąd jest niedopuszczalny.

Dlatego też dobry programista musi zwracać uwagę na typy zmiennych i ewentualną utratę precyzji przy obliczeniach.

Żeby nasz program działał prawidłowo należy użyć typu danych o odpowiedniej precyzji np. float.

```
float a = 5;
float b = 2;

Console.WriteLine(a / b);

Console.ReadLine();
```

Wprowadzanie danych do konsoli i parsowanie danych:

W trakcie działania programu konsolowego, często przydatne jest pobranie od użytkownika informacji np. do wypełnienia formularza.

Służy do tego metoda:

Console.ReadLine();

Pobiera ona od użytkownika ciąg znaków wpisywany z klawiatury i zwraca w postaci łańcucha znaków czyli typ string.

Skoro wiemy już, że Console.ReadLine() zwraca nam string, to co jeśli w naszym formularzu potrzebujemy np. uzupełnić zmienną typu int trzymającą w sobie np. wiek?

Powstaje problem, jak przerobić string na int.

Do zamiany typów służy tzw. parsowanie.

Przykład:

```
int wiek = int.Parse(Console.ReadLine());
```

Wiemy, że typem, którego oczekujemy jest int, więc na tym typie, wywołujemy metodę Parse(), która przerobi nam string z Console.ReadLine() na naszą zmienną typu int z wiekiem.

Klasa Math

Obsługa bardziej zaawansowanych działań i funkcji matematycznych, jak i podstawowe stałe matematyczne zostały zebrane w klasie Math.

Najważniejsze działania:

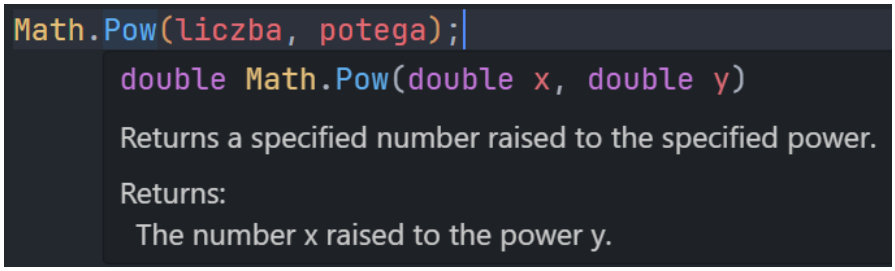
- **Potęgowanie** - służy do tego metoda **Pow()** od angielskiego power.

Zwraca ona liczbę podniesioną do określonej potęgi.

Przykład:

```
double liczba = 5;
double potega = 3;
double wynik = Math.Pow(liczba, potega);
Console.WriteLine(wynik);
```

Metoda Math.Pow() przyjmuje i zwraca jedynie liczby typu double



- **Pierwiastek kwadratowy** – Sqrt()

  Przykład:

```
double liczba = 9;
double wynik = Math.Sqrt(liczba);
Console.WriteLine(wynik);
```

Tak jak poprzednio pokazujemy, jak w VS sprawdzić przyjmowany i zwracany typ metody.

```
Math.Sqrt(liczba);
```

```
double Math.Sqrt(double d)

Returns the square root of a specified number.

Returns:
  One of the values in the following table.

d parameter – Return value
Zero or positive – The positive square root of d.
Negative –double.NaN
Equals double**.NaN –double.NaN
Equals double.**PositiveInfinity –double.PositiveInfinity
```

- **Stała Pi** – Oprócz metod, klasa Math posiada przydatne stałe matematyczne np:

```
Console.WriteLine(Math.PI);
```

Dodatkowa misja Programisty:

1. Napisz program, który będzie zwracać kwadrat dla wpisanej liczby np. dla 2 wyświetli się 4, dla 6 wyświetli się 36.
    a. przygotuj zmienną typu int do której zapiszesz liczbę wpisaną przez użytkownika, pamiętaj o prasowaniu
    b. przygotuj zmienną do której zapiszesz wynik odpowiednich obliczeń

c. wyświetl informację dla użytkownika np. po wpisaniu 2 powinno się wyświetlić:” 2 podniesione do potęgi 2 jest równe 4 “.