Lekcja 4
Funkcje

po co funkcje – struktura – parametry – zwracanie wartości - zasięg zmiennych – przekazywanie parametrów

Czym jest funkcja (procedura)? To zestaw instrukcji, które opisujemy jednym słowem, które porządkują strukturę programu przez podział na logiczne fragmenty, i które potem możemy wielokrotnie używać w programie głównym. Dzięki takiej organizacji łatwiej jest skracamy kod programu, porządkujemy go w logicznie uporządkowane fragmenty, możemy go szybciej poprawiać oraz możemy pisać biblioteki funkcji, z których będziemy mogli korzystać w innych programach. W bibliotekach języka C++ mamy mnóstwo gotowych funkcji, na przykład wykorzystywane już pow i sqrt z biblioteki cmath.

Struktura funkcji

typ nazwa (parametry){
instrukcje;
return wartość;
}

lub funkcja która nie zwraca żadnych wartości, tzw. procedura

void nazwa (parametry){ instrukcje;… }

Uwagi o funkcjach

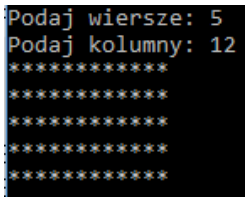
- funkcja powinna być zdefiniowana przed pierwszym jej wywołaniem
- funkcje definiujemy na zewnątrz funkcji main
- funkcje mogą zwracać wyliczone wartości lub nie
- funkcja „komunikuje” się z programem poprzez nazwę i parametry
- funkcja może nie mieć parametrów
- funkcja mini, która nic nie robi ***void funkcja() {};***

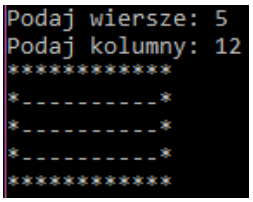
Zadanie - Gwiazdki

Napisz funkcję GWIAZDKI, która „rysuje” na ekranie prostokąt z gwiazdek. Parametry funkcji W i K – ile wierszy i kolumn. Dane wczytujemy z klawiatury w programie głównym. Zmodyfikuj program tak, aby środek prostokąta wypełniony był kropkami (brzeg z gwiazdek).

```
void GWIAZDKI(int W, int K){
  for (int i=1;i<=W;i++){
    for (int j=1;j<=K;j++)
       cout << "*";
    cout << endl;
  }
}
void GWIAZDKI1(int W, int K){
  for (int i=1;i<=W;i++){
    for (int j=1;j<=K;j++)
       if (i==1 || i==W || j==1 || j==K)
         cout << "*";
       else cout << "-";
    cout << endl;
}}
...
int wiersz,kolumna;
cout << "Podaj wiersze: ";
cin >> wiersz;
cout << "Podaj kolumny: ";
cin >> kolumna;
GWIAZDKI(wiersz,kolumna);
GWIAZDKI1(wiersz, kolumna);
```

- funkcja rysuje gwiazdki i nic nie zwraca, dlatego void
- wykonujemy funkcję podając jej nazwę i wstawiając parametry
- do funkcji wpisujemy parametry **wiersz** i **kolumna**, a funkcja po uruchomieniu pobiera te wartości i działa już na własnych **W** i **K**
- zwróć uwagę na wcięcia i wyrównanie bloków programu za pomocą nawiasów {}
- zewnętrzna pętla for wykonuje dwie instrukcje (for i cout), dlatego blok {}
- wewnętrzna instrukcja for wykonuje tylko jedną instrukcję warunkową if, dlatego nie musi być stosowany blok {}





Zadanie - Odległość

Oblicz odległość pomiędzy dwoma punktami A i B układu współrzędnych. Z klawiatury wpisujemy cztery wartości współrzędnych dwóch punktów A(x1,y1) i B(x2,y2).

```
float ODL(float x1, float y1, float x2, float y2){
      return sqrt(pow(x2-x1,2)+pow(y2-y1,2));
}
...
cout << ODL(1,1,3,3) << endl;
```

- funkcja oblicza odległość i zwraca wynik poprzez nazwę ODL
- wartość odległości wyświetlana jest na ekranie konsoli

UWAGA. Ponieważ używamy funkcji matematycznych **pow** i **sqrt**, dlatego należy zadeklarować bibliotekę **cmath**.

Zasięg zmiennych - zmienne globalne i lokalne

Programy się komplikują, zaczynamy używać wielu zmiennych i nazwy tych zmiennych mogą się powtarzać – będą takie same w programie głównym i w funkcjach. Bez problemu można definiować zmienne o takich samych nazwach, ale powinniśmy stosować kilka zasad:

- zmienne w funkcjach są lokalne, tworzone są w momencie wywołania funkcji i giną kiedy funkcja kończy działanie
- zmienne w programie głównym są globalne, widoczne są też w zdefiniowanych funkcjach
- zmienne zdefiniowane w funkcjach są ważniejsze niż takie same globalne
- żeby nie było niejasności, to lepiej definiować zmienne i stałe w funkcjach z różnymi nazwami, na przykład dodawać dodatkowy przedrostek

Zadanie – Silnia

Napisz funkcję SILNIA z jednym parametrem N, która zwraca wartość silni. Sprawdź, czy parametr jest liczbą naturalną z przedziału 1-100.
Sprawdź działanie funkcji wypisując na ekranie wartość 5!.
Sprawdź działanie funkcji wypisując na ekranie wartości kolejnych silni dla liczb 1-100.

```
int SILNIA(int N){
  int s=1;
  for (int i=1; i<=N; i++)
    s=s*i;
  return s;
}

int main(){
  cout << SILNIA(5) << endl;
}
```

Zadania

1) *Napisz funkcje obliczające: pole powierzchni i obwód: kwadratu, prostokąta, koła, trójkąta, sześciokąta foremnego, trapezu. Wprowadź dane z klawiatury i wylicz wartości za pomocą funkcji. Sprawdź czy parametry funkcji są większe zera.*

2) *Napisz funkcje obliczające: pole powierzchni i objętość kuli, prostopadłościanu, stożka, ostrosłupa. Wprowadź dane z klawiatury i wylicz wartości za pomocą funkcji. Sprawdź czy parametry funkcji są większe zera.*

3) *Napisz funkcję KWADRATOWA, która wymaga trzech parametrów rzeczywistych: a, b, c, i która wyliczy i wypisze na ekranie pierwiastki równania kwadratowego. Jeśli równanie nie ma rozwiązań – funkcja wypisuje komunikat „brak pierwiastków”.*

4) *Napisz funkcję ULAMEK z jednym parametrem X, która zwraca wynik poniższego ułamka. Wylicz wartość ułamka dla kolejnych liczb naturalnych w przedziale 1..100.*

$$X+\cfrac{X+\cfrac{X+\cfrac{X+\cfrac{X+X}{X}}{X}}{X}}{X}=$$

5) *Napisz funkcję o nazwie NEWTON z dwoma parametrami N i K, która w wyniku podaje wartość symbolu Newtona opisanego wzorem. Sprawdź, czy wpisane liczby są naturalne.*

$$\binom{n}{k}=\frac{n!}{k!(n-k)!}=\frac{n\cdot(n-1)\cdot\ldots\cdot(n-k+1)}{k!}$$