Część XVI C++ Funkcje

Jeśli nasz program rozrósł się już do kilkudziesięciu linijek, warto pomyśleć o jego podziale na mniejsze części. Poznajmy więc funkcje.

Szybko się przekonamy, że funkcja to bardzo pożyteczny twór języka programowania. Pozwala wydzielić fragment programu wykonujący określone zadanie i w ten sposób uporządkować kod źródłowy. Stosowanie funkcji ma też wiele innych zalet. Każdą funkcję możemy wywołać wielokrotnie w dowolnym miejscu naszego programu. Dzięki temu polecenia umieszczone wewnątrz funkcji w kodzie zapisywane są tylko raz, a wykonywane dowolną liczbę razy. Do funkcji możemy ponadto przesyłać różne dane i na ich podstawie wykonywać obliczenia, których wynik będzie zwracała funkcja. Na koniec omówimy zagadnienie ściśle związane z funkcjami - widoczność zmiennych. Przekonamy się, że niektóre zmienne mogą być w określonych miejscach kodu niewidoczne, z innych będziemy mogli natomiast korzystać w każdej chwili.

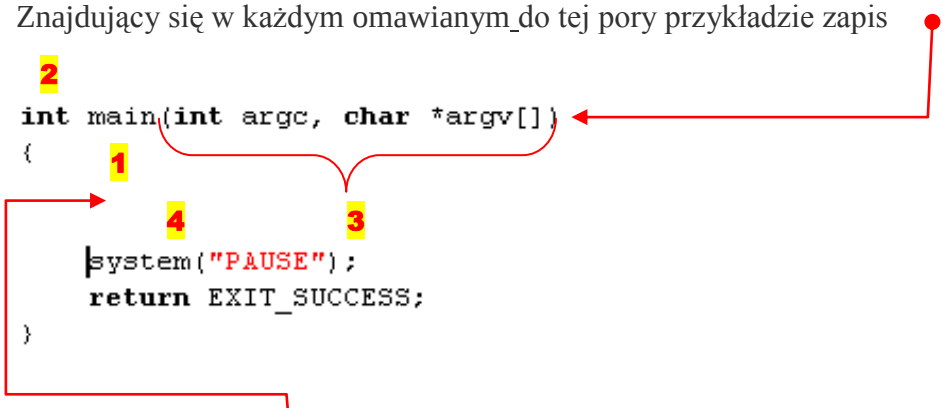
1.Co to jest funkcja ?

Poznane przez nas do tej pory konstrukcje pozwalały tworzyć programy liniowe. Dzięki funkcjom jeden fragment kodu będziemy mogli wykorzystać wielokrotnie.

Do tej pory tworzyliśmy różne bloki instrukcji. Bloki te były wykonywane warunkowo (jeśli wykorzystaliśmy instrukcje warunkowe) lub wielokrotnie (gdy blok instrukcji stanowił ciało pętli). Poznaliśmy też instrukcję goto, dzięki której można w programie przejść w zupełnie inne miejsce kodu i wykonać znajdujące się tam instrukcje. Nadszedł czas, aby poznać dosyć specyficzny sposób na wyodrębnianie w jeden blok niektórych instrukcji - nauczmy się posługiwać funkcjami. W poprzednich ćwiczeniach poznaliśmy już kilka funkcji. Korzystaliśmy z funkcji o nazwie **main**, czyli tak zwanej funkcji głównej. W ostatnim przykładzie znalazła się również inna funkcja – int().

2.Definicja funkcji

Znajdujący się w każdym omawianym do tej pory przykładzie zapis



(gdzie w miejscu ● znajdowały się różne instrukcje) to tak naprawdę definicja funkcji **main**.

W definicji każdej funkcji muszą się znaleźć następujące elementy:

- określenie nazwy funkcji 1 – funkcja główna musi się nazywać **main**,
- określenie typu danych będących wynikiem działania funkcji 2 - funkcja **main** powinna zwracać liczbę całkowitą (a najlepiej liczbę 0 - wtedy dla systemu jest to informacja, że wykonanie funkcji głównej zostało zakończone sukcesem),
- argumenty (wraz z ich typami) funkcji 3, czyli wartości przekazywane do wnętrza funkcji - do funkcji **main** przekazywane są dwa argumenty,
- instrukcje wykonywane wewnątrz funkcji 4 - dla funkcji głównej dotychczas instrukcje te stanowiły cały kod naszego programu,
- instrukcję **return** , dzięki której jako wynik działania funkcji zwracana jest wartość zmiennej lub

wyrażenia znajdującego się po słowie kluczowym **return** - funkcja główna zwraca wartość **EXIT SUCCESS**, która jest równoważna liczbie 0.

3.Wywołanie funkcji

Wiemy już ogólnie, jak zdefiniować funkcję. Ale po co to robić? Na przykład po to, aby ją w dowolnym miejscu programu wywołać, przekazując do niej dowolny jeden lub więcej argumentów (oczywiście takiego typu, jaki ustaliliśmy w definicji) . Funkcja **main** jest dosyć specyficznym rodzajem funkcji - nie trzeba jej wywoływać, bo od niej zaczyna się wykonywanie naszego programu. Stwórzmy więc własną, bardzo prostą funkcję - najpierw ją zdefiniujemy, a potem wywołamy wewnątrz funkcji **main.**

Przy nazywaniu funkcji stosujemy zasady podobne do tych, którymi należy się kierować przy nadawaniu nazw zmiennym.

Ćwiczenie 1

a) Ustalamy, że nasza funkcja będzie nazywała się **dodawanie ()** . Będziemy do niej przekazywać dwie liczby całkowite, a efektem działania funkcji będzie suma tych liczb (wynik będzie również liczbą całkowitą). Tuż przed definicją funkcji **main** wpisujemy typ zwracanego przez naszą funkcję wyniku (**int** - oznacza, jak pamiętamy, typ całkowitoliczbowy) i wpisujemy nazwę funkcji

```
int dodawanie(int a, int b)
{
    return a+b;
}

int main(int argc, char *argv[])
{
    cout << "10 + 25 = ";
    cout << dodawanie(10,25) << endl;
```

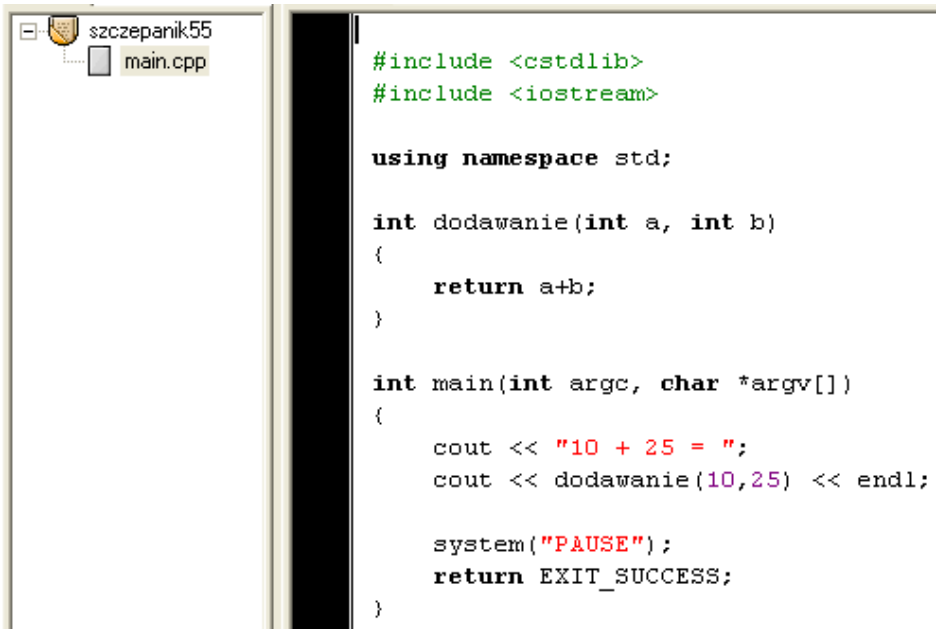
b) w nawiasach określamy argumenty, które zostaną przekazane do funkcji. Ustaliliśmy, że będą to dwie liczby całkowite -wpisujemy więc (nasze zmienne wejściowe nazywamy w dowolny sposób - oczywiście zgodnie z zasadami nadawania nazwy zmiennym).

c) teraz wystarczy już tylko wewnątrz nawiasów klamrowych umieścić instrukcje, które zostaną wykonane przez funkcję. Ponieważ nasza funkcja **dodawanie ()** jest niezwykle prosta, wpisujemy od razu słowo kluczowe **return** i wyrażenie . Dzięki temu funkcja zwraca sumę przekazanych do niej dwóch liczb.

d) aby się przekonać o tym, że nasza funkcja naprawdę działa, w funkcji głównej umieszczamy wywołanie, przekazując jako argumenty dwie liczby - 10 i 25.

e) utwórz nowy projekt w Dev C++ i zapisz go na pulpicie w folderze o nazwie **nazwisko55**

f) wprowadź do projektu modyfikacje tak aby wyglądał jak poniżej



g) z menu Uruchom wybieramy Kompiluj i uruchom

h) powinniśmy zobaczyć

Oczywiście, do funkcji, zamiast liczb, możemy przekazać również wartość zmiennej . Przekazywane zmienne mogą mieć taką samą nazwę, jak w definicji naszej funkcji. Pamiętajmy jednak, że mimo iż są tak samo nazwane, to w rzeczywistości są to różne zmienne. Nie ma również przeszkód, aby naszą funkcję wywołać w programie kilkakrotnie, przekazując do niej za każdym razem różne argumenty.

```
int x=23;
int b=7;

cout << x << " + " << b << " = " ;
cout << dodawanie(x,b) << endl;
```

4. Definicja a deklaracja funkcji

W poprzednim przykładzie umieściliśmy definicję funkcji **dodawanie()** tuż przed funkcją **main**. Sprawdźmy, co się stanie, jeśli zamienimy definicje tych funkcji miejscami .

Ćwiczenie nr 2

1. Utwórz nowy projekt w Dev C++ i zapisz go na pulpicie w folderze o nazwie **nazwisko56**
2. Wprowadź do projektu modyfikacje tak aby wyglądał jak poniżej

```
#include <cstdlib>
#include <iostream>

using namespace std;

int main(int argc, char *argv[])
{
     cout << "10 + 25 = ";
     cout << dodawanie(10,25) << endl;

     system("PAUSE");
     return EXIT_SUCCESS;
}
int dodawanie(int a, int b)
{
     return a+b;
}
```

3. Z menu Uruchom wybieramy Kompiluj i uruchom
Próba skompilowania takiego programu kończy się wystąpieniem błędu.
Dlaczego tak się dzieje? Odpowiedź jest prosta. Wywołanie funkcji **dodawanie()** wystąpiło przed jej definicją - to tak, jakbyśmy chcieli zapisać coś do zmiennej, której wcześniej nie zadeklarowaliśmy.

4. Nie poprawiaj kodu programu – zamknij program i przejdź do dalszej części

5. Czy więc skazani jesteśmy na definiowanie własnych funkcji przed funkcją główną? Na szczęście nie - byłoby to bardzo niewygodne, bo analizę programu należy zawsze rozpoczynać od funkcji głównej, która w takim wypadku znajdowałaby się gdzieś na końcu kodu. Jak więc rozwiązać ten problem? Naszą funkcję możemy umieścić poniżej funkcji głównej, ale przed pierwszym wywołaniem naszej funkcji należy ją zadeklarować

```
#include <cstdlib>
#include <iostream>

using namespace std;

int dodawanie(int a, int b);

int main(int argc, char *argv[])
{
     cout << "10 + 25 = ";
     cout << dodawanie(10,25) << endl;

     system("PAUSE");
     return EXIT_SUCCESS;
}

int dodawanie(int a, int b)
{
     return a+b;
}
```

Deklaracja funkcji od jej definicji różni się tym, że w deklaracji, zamiast wpisywania ciała funkcji, umieszczamy jedynie średnik.

Deklaracja jest więc tylko zapowiedzią naszej funkcji.

Kompilator na podstawie deklaracji wie, jaki typ wyniku zwraca funkcja oraz ile oraz jakiego typu argumenty należy do niej przekazać. Na tej podstawie może już sprawdzić, czy wywołanie naszej funkcji jest prawidłowe.

Co prawda deklaracja funkcji może nastąpić tuż przed jej wywołaniem, jednak zalecane jest umieszczanie deklaracji funkcji przed funkcją główną. Dzięki temu kod jest czytelniejszy i łatwiej wprowadzać do niego zmiany.

Ćwiczenie nr 3

1. Utwórz nowy projekt w Dev C++ i zapisz go na pulpicie w folderze o nazwie **nazwisko57**
2. Wprowadź do projektu modyfikacje tak aby wyglądał jak poniżej

```
#include <cstdlib>
#include <iostream>

using namespace std;

int dodawanie(int a, int b);

int main(int argc, char *argv[])
{
    cout << "10 + 25 = ";
    cout << dodawanie(10,25) << endl;

    system("PAUSE");
    return EXIT_SUCCESS;
}

int dodawanie(int a, int b)
{
    return a+b;
}
```

3. z menu Uruchom wybieramy Kompiluj i uruchom.

6. Argumenty funkcji głównej

W każdym napisanym przez nas programie w definicji funkcji **main** podawaliśmy (a raczej były generowane przez - Dev-C++) dwa argumenty

```
int main(int argc, char *argv[])
{
    cout << "Liczba argumentow: " << argc << endl;
    int i = 0;
    for (int i=0; i < argc; i++)
    {
        cout << i << " -> " << argv[i] << endl;
    }

    cout << endl;

    system("PAUSE");
    return EXIT_SUCCESS;
}
```

W zmiennej całkowitoliczbowej **argc** znajduje się liczba przekazanych do funkcji **main** (czyli naszego programu) argumentów. Zmienna ***argv** to tablica (tę konstrukcję języka poznamy dalej), w której znajdują się te argumenty.

W linii wyświetlamy liczbę argumentów przekazanych do funkcji.

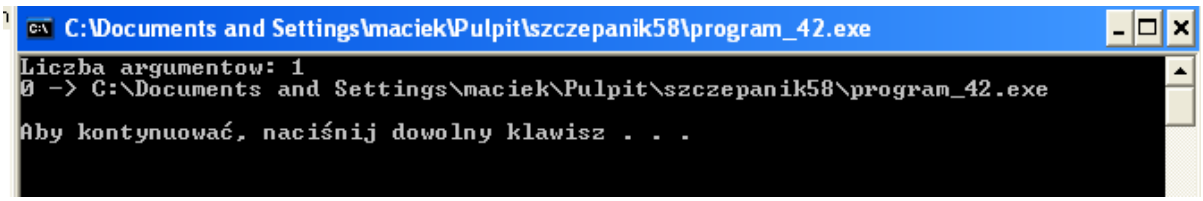
Następnie w pętli wyświetlamy kolejne argumenty. Dla ułatwienia przed każdym argumentem wyświetlamy jego numer oraz i strzałkę.

Ćwiczenie nr 4

1. Utwórz nowy projekt w Dev C++ i zapisz go na pulpicie w folderze o nazwie **nazwisko58**
2. Wprowadź do projektu modyfikacje tak aby wyglądał jak poniżej

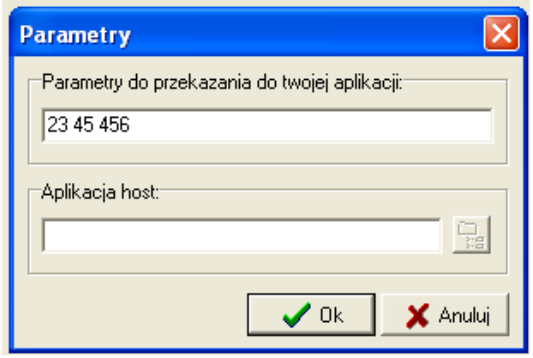
```
#include <cstdlib>
#include <iostream>

using namespace std;

int main(int argc, char *argv[])
{
    cout << "Liczba argumentow: " << argc << endl;
    int i = 0;
    for (int i=0; i < argc; i++)
    {
        cout << i << " -> " << argv[i] << endl;
    }

    cout << endl;

    system("PAUSE");
    return EXIT_SUCCESS;
}
```

3. Skompiluj i uruchom programu
4. Na ekranie pojawi się widok

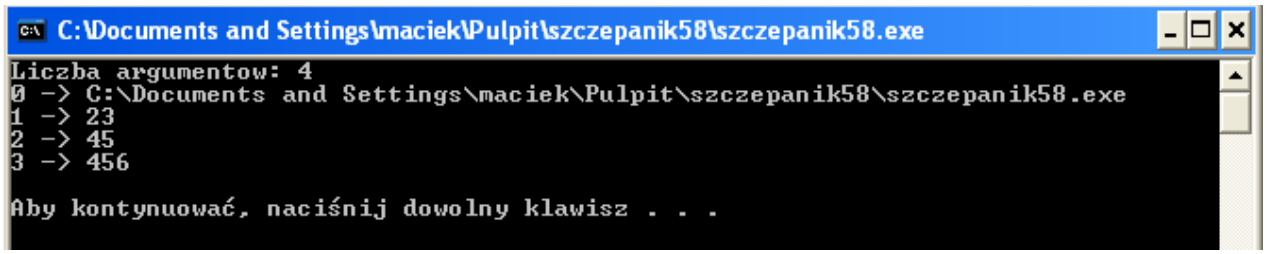


Jak łatwo zauważyć, do funkcji **main** został przekazany jeden argument(o numerze 0), którego wartością jest ścieżka i nazwa naszego programu. Ten argument jest zawsze dostarczany do każdego programu.

7. Argumenty do funkcji głównej możemy przekazać również z poziomu aplikacji Dev-C++.
Aby to zrobić, należy:
- w folderze nazwisko58 uruchomić plik o nazwie **nazwisko58.dev**
-z menu Uruchom wybieramy pozycję Parametry... i w pole wpisujemy argumenty.



-po kliknięciu na OK kompilujemy i uruchamiamy program - argumenty zostaną przekazane.
- powinniśmy zobaczyć



Wszystkie pliki z nazwiskiem i kolejnym numerem umieszczamy w swoim folderze nazwiskocplusplus na serwerze.