Część XI C++

W folderze nazwisko36 program za każdym razem sprawdza oba warunki co niepotrzebnie obciąża procesor. Ten problem można rozwiązać stosując instrukcje **if...else**
Instrukcja if wykonuje polecenie tylko w przypadku spełnienia określonego warunku. Gdy warunek nie był spełniony , nie było wykonywane żadne polecenie.
Dzięki modyfikacji instrukcji if możemy dodatkowo określić polecenia, które zostaną wykonane gdy warunek nie został spełniony.
Taką instrukcje warunkową określamy jako if...else

Ogólna postać instrukcji if else wygląda tak:

```
if (warunek){
  instrukcja1;
...
  instrukcjaN;
} else {
  instrukcjaA;
...
  instrukcjaZ;
}
```

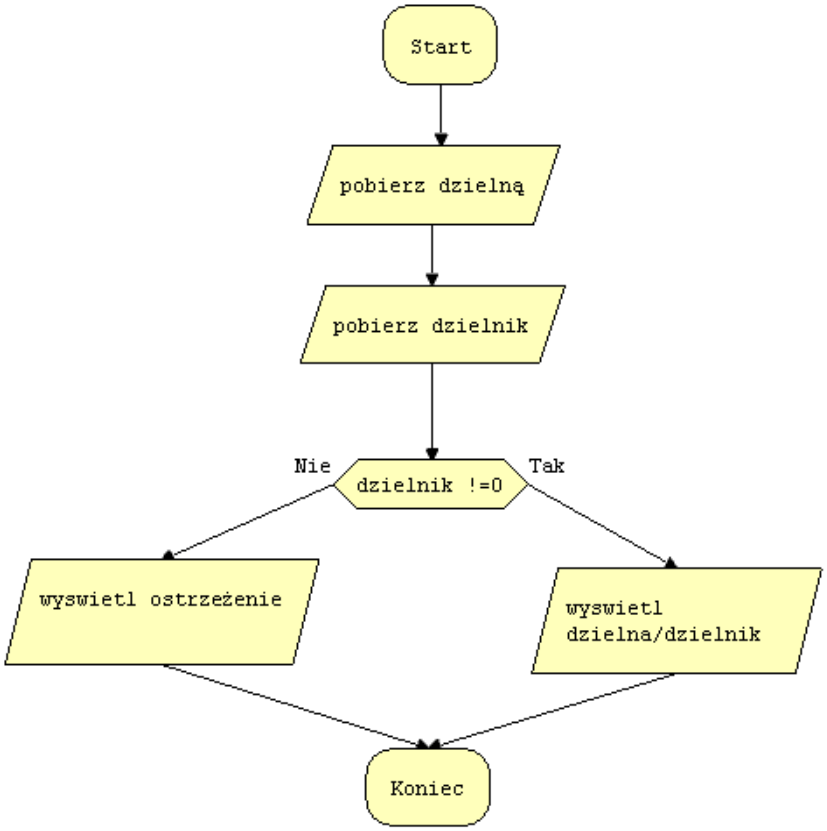
ZAPIS CZYTAMY TAK:
Jeśli warunek jest spełniony wykonaj instrukcję z pierwszego bloku

W przeciwnym wypadku wykonaj instrukcję z bloku drugiego

Jeśli w wypadku spełnienia warunku ma być wykonana tylko jedna instrukcja, nie musimy jej umieszczać w nawiasach klamrowych. Podobnie w wypadku niespełnienia warunku.

Ćwiczenie 1 – utworzyć program dzielący przez siebie dwie liczby

1. Utwórz schemat blokowy (według poniższego wzoru) i zapisz go w folderze **nazwisko37**



2. Utwórz nowy projekt w Dev C++ i zapisz go w folderze **nazwisko37**
3. Wprowadź do projektu modyfikacje tak aby wyglądał jak poniżej
4. Skompiluj i uruchom program
5. Przeanalizuj program

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

using namespace std;

int main(int argc, char *argv[])
{
    float dzielna, dzielnik;
    cout << "Podaj dzielna : ";
    cin >> dzielna;
    cout << "Podaj dzielnik : "
    cin >> dzielnik;

    if ( dzielnik != 0) {
         cout << "Iloraz podanych liczb wynosi: ";
         cout << dzielna / dzielnik << endl;
    }
    else cout << "Nie mozna dzielic przez zero!" << endl;


    cout << endl << endl;

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

Za pomocą instrukcji warunkowej sprawdzamy czy dzielnik jest różny od zera

Jeśli dzielnik jest różny od zera wyświetlamy wynik dzielenia

Jeśli dzielnik jest równy zero wyświetlamy komunikat

Wielokrotny warunek

Potrafimy już niektóre instrukcje wykonać tylko wtedy, gdy spełniony jest określony warunek, a inne gdy warunek nie jest prawdziwy. Jednak czasem zachodzi potrzeba wykonania kilku różnych poleceń - każdego przy spełnieniu innego warunku.

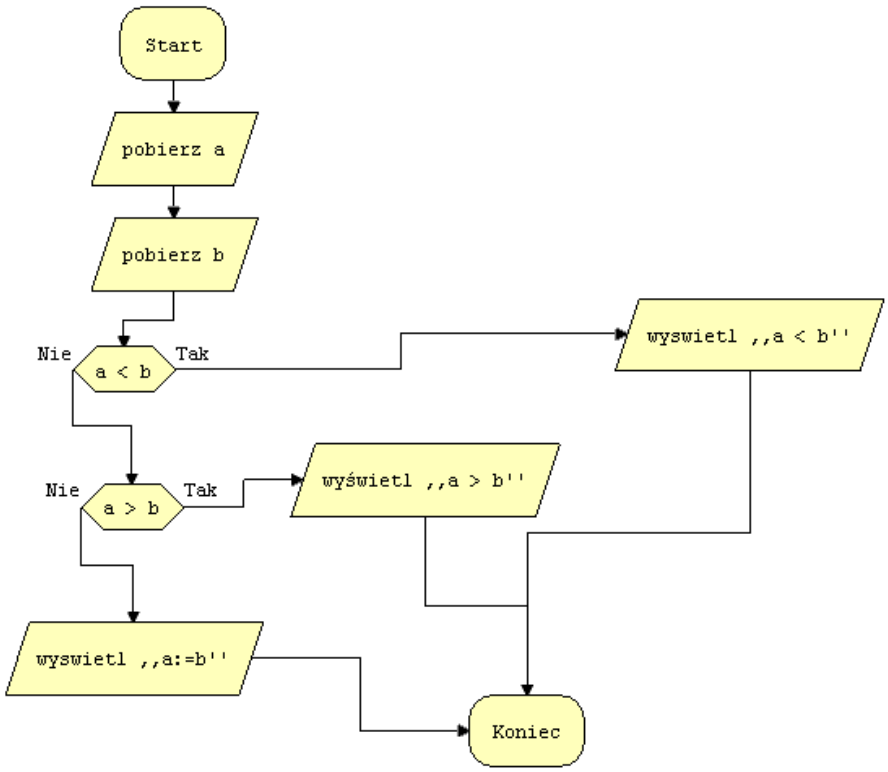
Taki przykład wykonaliśmy za pomocą instrukcji if w programie w folderze nazwisko35. Otwórz folder i sprawdź uruchamiając plik z rozszerzeniem DEV - zastosowaliśmy po trzy instrukcje warunkowe if.

Nie jest to jednak rozwiązanie zbyt eleganckie - jeśli nawet został spełniony warunek z pierwszej instrukcji if i wiadomo, że warunki z kolejnych instrukcji if nie mogą być prawdziwe, i tak są one sprawdzane, co zajmuje niepotrzebnie czas procesora.

Dużo lepiej jest w takim wypadku zastosować instrukcję wielokrotnego warunku określaną jako if...else if.

Ćwiczenie 2

1. Utwórz folder **nazwisko 38** i zapisz w nim schemat blokowy według poniższego wzoru



2. **Skopiuj zawartość folderu nazwisko35 do folderu nazwisko38** .
3. Następnie w folderze **nazwisko38 usuń plik nazwisko35.exe** (prostokątna ikona z trzema białymi kółeczkami)
4. Zmień nazwę pliku **nazwisko35.DEV** na **nazwisko38.DEV**
5. Uruchom plik **nazwisko38.DEV**
6. Wprowadź modyfikacje według poniższego wzoru

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

using namespace std;

int main(int argc, char *argv[])
{
    float a, b;
    cout << "Podaj liczbe A : ";
    cin >> a;
    cout << "Podaj liczbe B : ";
    cin >> b;

    if ( a < b ) cout << "Liczba A jest mniejsza od B." << endl;
    else if ( a > b ) cout << "Liczba A jest wieksza od B." << endl;
    else cout << "Liczby A i B sa rowne." << endl;


    cout << endl << endl;

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

7. Skompiluj i uruchom program
8. Przeanalizuj program!!!

Instrukcja wyboru switch

Poznana przed chwilą instrukcja wielokrotnego warunku jest dosyć skomplikowaną strukturą - łatwo się w niej pogubić i trudno ją zmodyfikować. Przedstawiony przykład jej zastosowania można dosyć łatwo uprościć i sprawić, że kod będzie wyglądał przejrzyście. Wszystko dzięki instrukcji wyboru switch.

Jej ogólna postać wygląda następująco:

```
switch (zmienna lub wyrażenie) {
case liczba1: instrukcje1; break;
case liczba2: instrukcje2; break;
...
case liczbaN: instrukcjeN; break;
default: instrukcje domyślne;
}
```

Polecenie break powoduje przerwanie działania instrukcji switch. Powinno się ono znajdować na końcu instrukcji występujących po każdym słowie case. Występowanie po instrukcjach domyślnych (występujących po słowie kluczowym default) nie jest obowiązkowe (instrukcja switch bowiem i tak się po nich kończy).

Wartość wyrażenia lub zmiennej powinna być typu całkowitoliczbowego lub znakowego. Wtedy zamiast liczb po słowie kluczowym case mogą wystąpić symbole znakowe.

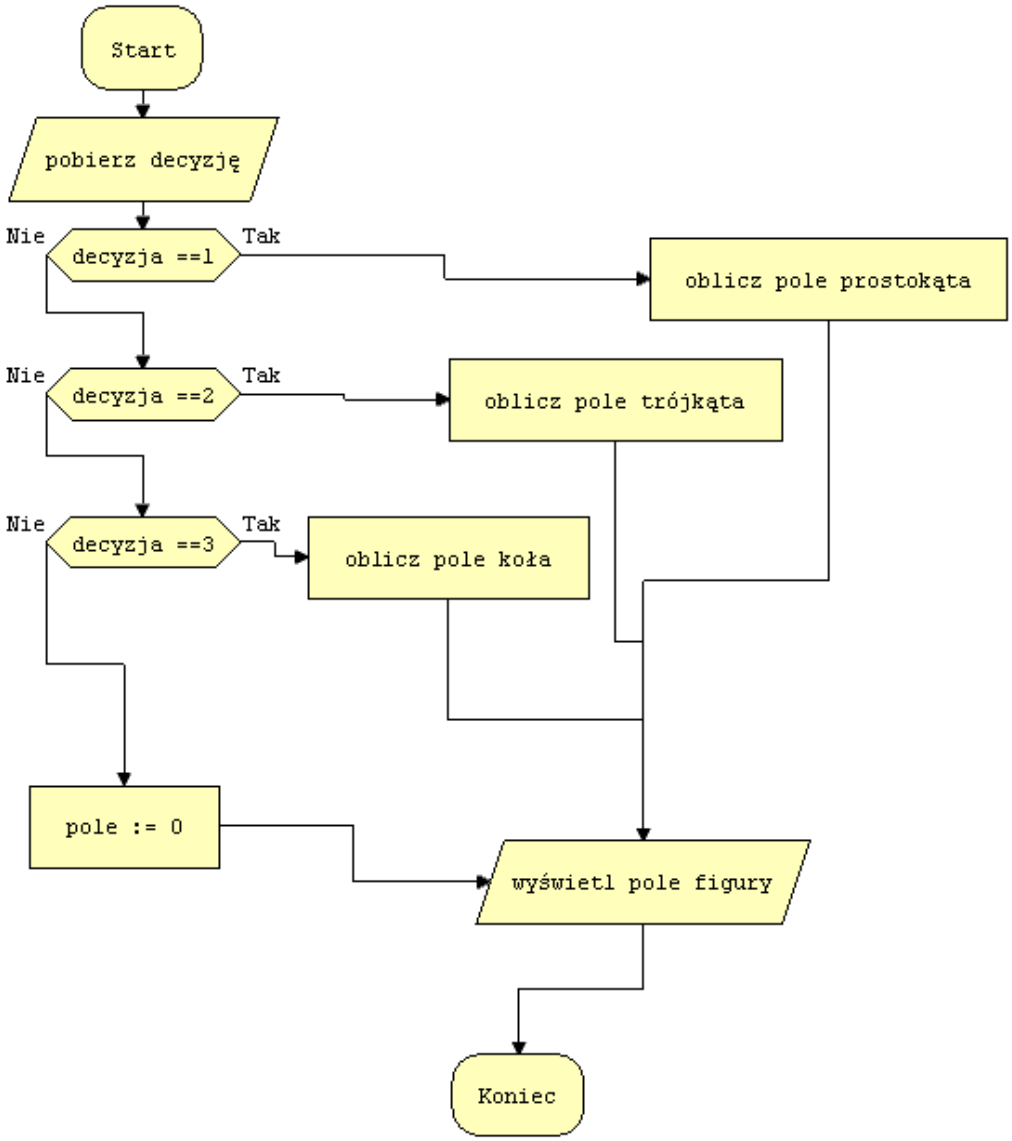
Działanie instrukcji wyboru switch można opisać w taki sposób:

- ✓ jeśli w nawiasie po słowie kluczowym switch znajduje się wyrażenie, jest ono obliczane,
- ✓ gdy wartość zmiennej lub obliczonego wyrażenia równa jest wartości liczbal, wykonywane są instrukcje1, gdy wartość ta wynosi liczba2, wykonywane są instrukcje2 i tak dalej,
- ✓ gdy nie uda się dopasować wartości zmiennej (lub obliczonego wyrażenia) do żadnej wartości występującej po słowie case, wykonywane są instrukcje domyślne występujące po słowie kluczowym default.

Ćwiczenie 3

Cel ćwiczenia: napisać zmodyfikowany program obliczający pola różnych figur.

1.Utwórz folder **nazwisko39**
2.Utwórz schemat blokowy według poniższego wzoru i zapisz w folderze **nazwisko39**



3.Utwórz nowy projekt w Dev C++ i zapisz go w folderze **nazwisko39**
4.Wprowadź do projektu modyfikacje tak aby wyglądał jak poniżej
5.Wprowadzając zmiany **czytaj analizę** programu w żółtych ramkach

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

using namespace std;

int main(int argc, char *argv[])
{
    float x, y, pole;
    int decyzja;
    cout << "Pole jakiej figury chcesz obliczyć? Podaj numer figury." << endl;
    cout << "1 - prostokat" << endl << "2 - trojkat" << endl << "3 - kolo" << endl;
    cin >> decyzja;

    switch (decyzja){
           case 1:
                cout << "Podaj dlugosci bokow: ";
                cin >> x >> y;
                pole = x * y;
                break;
           case 2:
                cout << "Podaj dlugosc podstawy i wysokosci: ";
                cin >> x >> y;
                pole = 0.5 * x * y;
                break;
           case 3:
                cout << "Podaj promien: ";
                cin >> x;
                pole = 3.14 * x * x;
                break;
           default:
                cout << "Zly wybor. Koniec programu" << endl;;
                pole = 0;
    }
    cout << "Pole figury wynosi " << pole << endl;

    cout << endl << endl;
    system("PAUSE");
    return EXIT_SUCCESS;
}
```

Na początku definiujemy trzy zmienne

wyświetlamy komunikat z prośbą o wybór figury, dla której zostanie obliczone pole

Umieszczamy instrukcję switch

Zależnie od wartości zmiennej „decyzja" wykonywane są bloki instrukcji: 1 gdy decyzja ma wartość 1.

Gdy decyzja wynosi 2

Gdy decyzja wynosi 3

Na końcu umieszczamy słowo kluczowe default Instrukcje , które się po nim znajdują, zostaną wykonane wtedy, gdy użytkownik poda inną wartość niż liczba od 1 do 3.

Po umieszczeniu nawiasu klamrowego kończącego instrukcje switch, wyświetlamy obliczone pole figury

5.Skompiluj i uruchom program

Instrukcja wyboru switch wydaje się doskonałym sposobem na stworzenie programu składającego się z kilku części (tak jak w wymienionym przykładzie). Pamiętajmy jednak, aby nie tworzyć zbyt rozbudowanych instrukcji switch. Kod wtedy stanie się niezbyt przejrzysty i bardzo trudno modyfikowalny. Zamiast skomplikowanych instrukcji switch możemy skorzystać na przykład z funkcji.

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