## Część XIV C++

### Złożone wyrażenia w pętli for

W wypadku zagnieżdżenia dwóch pętli druga pętla wykonywana jest w całości w każdym przebiegu pętli pierwszej. Jednak niekiedy zachodzi potrzeba równoczesnego wykonywania dwóch pętli. Taki efekt można uzyskać, stosując złożone wyrażenia w nagłówku pętli.
Spójrzmy na przykład

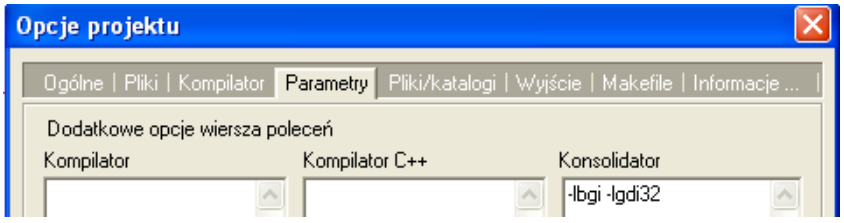
```
for (int x = 0, r = 50; x < getmaxx(), r > 0; x += 50, r -= 5)
{
        delay(50);
        circle(x,50,r);
}
```

w pierwszej pętli do każdego wyrażenia (inicjującego, warunkowego, zmieniającego) po przecinku dodano odpowiednie wyrażenie z pętli drugiej. Program będzie działał w taki sposób:
- ►zostaną zainicjowane zmienne **x** i **r** - pierwszej zostanie przypisana wartość 0, drugiej - 50,
- ►ponieważ oba wyrażenia warunkowe będą w tej chwili spełnione, zostaną wykonane instrukcje , czyli zostanie narysowany okrąg o promieniu 50, którego środek znajduje się w punkcie 0,50.

- ► po wykonaniu instrukcji
```
delay(50);
circle(x,50,r);
```
wartość zmiennej **x** zostanie zwiększona o 50,a wartość zmiennej r zmniejszona o 5.
- ► w kolejnych przebiegach pętli zostaną narysowane okręgi: o środku 50,50 i promieniu 45, o środku 100,50 i promieniu 40 i tak dalej.

### Ćwiczenie 1

1. Utwórz nowy projekt w Dev C++ i zapisz go na pulpicie w folderze o nazwie **nazwisko48**
2. Z menu Projekt wybieramy Opcje projektu i w oknie klikamy na zakładkę Parametry W pole konsolidator wpisujemy: -lbgi -lgdi32 i zatwierdzamy Ok




3. Z menu Projekt wybieramy Dodaj do projektu,
4. Przechodzimy do Mój komputer →Dysk lokalny C → folder Dev-Cpp →folder include (katalogc:\dev-cpp\include) i dwukrotnie klikamy na ikonę winbgim 

5. Wprowadź do projektu modyfikacje tak aby wyglądał jak poniżej

```
#include <cstdlib>
#include <iostream>
#include <winbgim.h>

using namespace std;

int main(int argc, char *argv[])
{
    initwindow(400,300);
    setfillstyle(SOLID_FILL, BLUE);
    bar (0,0,getmaxx(),getmaxy());


    for (int x = 0, r = 50; x < getmaxx(), r > 0; x += 50, r -= 5)
    {
            delay(50);
            circle(x,50,r);
    }


    while(!kbhit());
    closegraph();


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

6. Z menu Uruchom wybieramy Kompiluj i uruchom
7. W oknie zapisz plik wskazujemy pulpit → folder nazwisko48 → otwieramy folder nazwisko48→ klikamy zapisz

**Niebezpieczeństwo złożonych warunków**

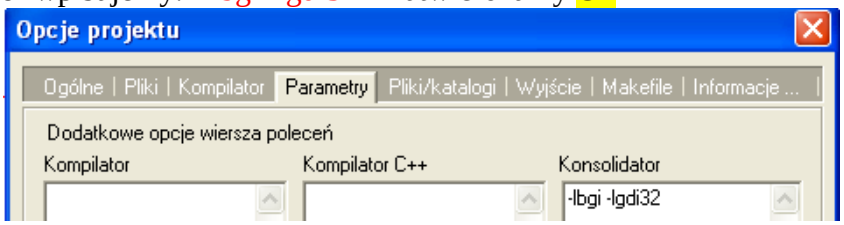
Złożone wyrażenia warunkowe w definicji pętli for stanowią pewne zagrożenie. Spójrzmy na przykład kodu , w którym występuje takie niebezpieczeństwo.

```
for (int r = 50, kolor = 0; r > 0, kolor <= 15; r -= 5, kolor++)
{
        delay(50); (1)
        setcolor(kolor); (2)
        circle(200,200,r); (3)
        cout << r << "-" << kolor << endl;
}
```

- Wykonanie pętli rozpoczyna się od zadeklarowania zmiennych **r** i **kolor** oraz zainicjowania ich wartościami 50 i 0.
- W ciele pętli oprócz opóźnienia **(1)** ustawiamy kolor o wartości zapisanej w zmiennej **kolor (2)** i w punkcie o współrzędnych 200,200 rysujemy okrąg o promieniu **r (3)** . Dla naszej informacji wyświetlamy jeszcze wartość obu zmiennych .
- Instrukcje z ciała pętli będą wykonywane, dopóki spełniane będą warunki **r>0, kolor<=15** (czyli gdy promień będzie większy od 0, a kolor mniejszy lub równy 15, przy czym promień w każdym przebiegu pętli zmniejszamy o 5, a zmienną kolor zwiększamy o1.

**Ćwiczenie nr 2**

1. Utwórz nowy projekt w Dev C++ i zapisz go na pulpicie w folderze o nazwie **nazwisko49**
2. Z menu Projekt wybieramy Opcje projektu i w oknie klikamy na zakładkę Parametry W pole konsolidator wpisujemy: -lbgi -lgdi32 i zatwierdzamy Ok

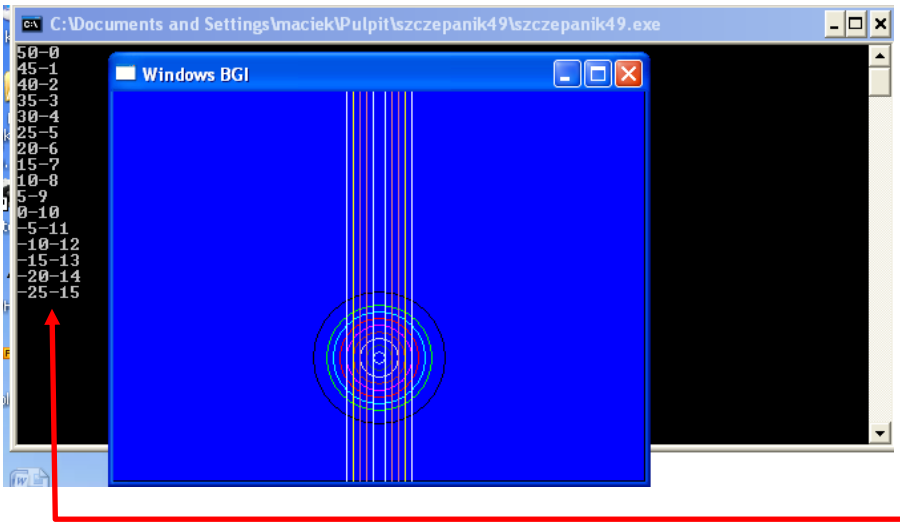



3. Z menu Projekt wybieramy Dodaj do projektu,
4. Przechodzimy do Mój komputer →Dysk lokalny C → folder Dev-Cpp →folder include (katalogc:\dev-cpp\include) i dwukrotnie klikamy na ikonę winbgim


5. Wprowadź do projektu modyfikacje tak aby wyglądał jak poniżej

```
#include <cstdlib>
#include <iostream>
#include <winbgim.h>

using namespace std;

int main(int argc, char *argv[])
{
    initwindow(400,300);
    setfillstyle(SOLID_FILL, BLUE);
    bar (0,0,getmaxx(),getmaxy());


    for (int r = 50, kolor = 0; r > 0, kolor <= 15; r -= 5, kolor++)
    {
            delay(50);
            setcolor(kolor);
            circle(200,200,r);
            cout << r << "-" << kolor << endl;
    }


    cout << endl << endl;
    while(!kbhit());
    closegraph();


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

6. Z menu Uruchom wybieramy Kompiluj i uruchom
7. W oknie zapisz plik wskazujemy pulpit → folder nazwisko49→ otwieramy folder nazwisko49→ klikamy zapisz



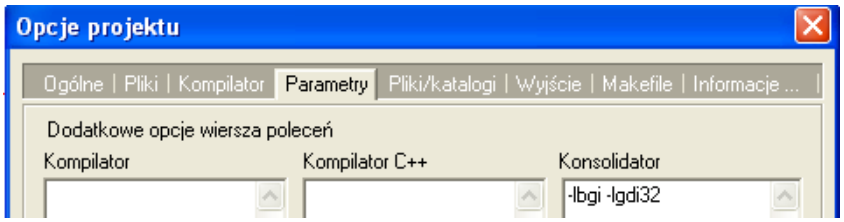

Na ekranie widzimy dziwny efekt . Dlaczego tak się dzieje wystarczy zobaczyć **wartości zmiennych** w kolejnych przebiegach. Jak widzimy, promień w pewnym momencie przyjmował wartości ujemne mimo warunku przejścia do następnej iteracji r > 0. Dlaczego tak się stało? Ponieważ pętla była wykonywana aż do momentu, w którym wszystkie wyrażenia warunkowe pętli stały się fałszywe.

Jeśli więc chcemy, aby pętla została zakończona wtedy, gdy którekolwiek z wyrażeń warunkowych zacznie zwracać wartość 0 (fałsz), zastosujmy operator iloczynu logicznego (jak pamiętamy, zwraca on wartość 1 tylko i wyłącznie wtedy, gdy oba wyrażenia zwracają wartość 1). Po takiej modyfikacji kodu efekt działania programu będzie całkiem inny.

## Ćwiczenie 3

1. Utwórz nowy projekt w Dev C++ i zapisz go na pulpicie w folderze o nazwie **nazwisko50**

2. Z menu Projekt wybieramy Opcje projektu i w oknie klikamy na zakładkę Parametry W pole konsolidator wpisujemy: -lbgi -lgdi32 i zatwierdzamy Ok.




3. Z menu Projekt wybieramy Dodaj do projektu,

4. Przechodzimy do Mój komputer →Dysk lokalny C → folder Dev-Cpp →folder include (katalogc:\dev-cpp\include) i dwukrotnie klikamy na ikonę winbgim 


5. Wprowadź do projektu modyfikacje tak aby wyglądał jak poniżej

```
#include <cstdlib>
#include <iostream>
#include <winbgim.h>

using namespace std;

int main(int argc, char *argv[])
{
    initwindow(400,300);
    setfillstyle(SOLID_FILL, BLUE);
    bar (0,0,getmaxx(),getmaxy());


    for (int r = 50, kolor = 0; (r > 0) && (kolor <= 15); r -= 5, kolor++)
    {
            delay(50);
            setcolor(kolor);
            circle(200,200,r);
            cout << r << "-" << kolor << endl;
    }

    cout << endl << endl;
    while(!kbhit());
    closegraph();


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

6. Z menu Uruchom wybieramy Kompiluj i uruchom
7. W oknie zapisz plik wskazujemy pulpit → folder nazwisko50 → otwieramy folder nazwisko50→ klikamy zapisz

**Jak przerwać działanie pętli - instrukcja break**

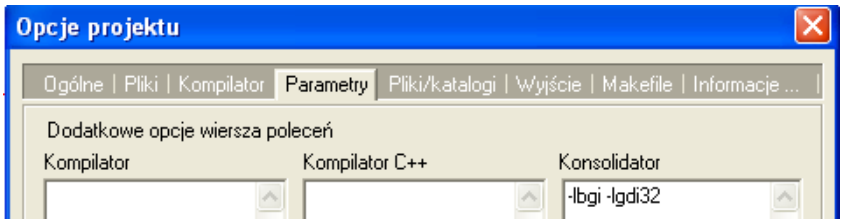
Poznaliśmy zasadę działania pętli programowej. Dowiedzieliśmy się, że dopóki spełniony jest określony warunek, instrukcje wewnątrz pętli wykonywane są „w kółko". Możliwe jest jednak przerwanie działania pętli niezależnie od wyrażenia warunkowego. Do tego celu służy instrukcja break. Korzystaliśmy już z niej przy instrukcji wyboru switch, gdzie również służyła do przerywania działania instrukcji, w której ją umieszczono. Spójrzmy na przykład.

```
int r = 5;
while (1) /*definiujemy zmienną r(przypisujemy jej wartośc 5)
      i tworzymy pętlę nieskończoną while */
{
      if (r>100) break;
      delay(10);
      circle(200,150,r);
      /* instrukcja rysuje w punkcie o współrzędnych 200,150
      okrąg o promieniu równym wartości zmiennej r, która w
      każdym przebiegu jest zwiększona o 5 (poniżej)*/
      r+=5;
```

Instrukcja warunkowa sprawdza, czy promień jest już większy od 100. Jeśli tak jest, wykonywana jest instrukcja break powodująca przerwanie działania pętli. Gdybyśmy nie wpisali tej instrukcji, na ekranie zobaczylibyśmy a program by się zawiesił. Dzięki instrukcji break możemy w dowolnym momencie przerwać działanie pętli.

## Ćwiczenie 4

1. Utwórz nowy projekt w Dev C++ i zapisz go na pulpicie w folderze o nazwie **nazwisko51**
2. Z menu Projekt wybieramy Opcje projektu i w oknie klikamy na zakładkę Parametry W pole konsolidator wpisujemy: -lbgi -lgdi32 i zatwierdzamy Ok




3. Z menu Projekt wybieramy Dodaj do projektu,
4. Przechodzimy do Mój komputer →Dysk lokalny C → folder Dev-Cpp →folder include (katalogc:\dev-cpp\include) i dwukrotnie klikamy na ikonę winbgim 

5. Wprowadź do projektu modyfikacje tak aby wyglądał jak poniżej

```
#include <cstdlib>
#include <iostream>
#include <winbgim.h>

using namespace std;

int main(int argc, char *argv[])
{
    initwindow(400,300);
    setfillstyle(SOLID_FILL, BLUE);
    bar (0,0,getmaxx(),getmaxy());

    int r = 5;
    while (1)
    {
          if (r>100) break;
          delay(10);
          circle(200,150,r);
          r+=5;
    }

    cout << endl << endl;
    while(!kbhit());
    closegraph();


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

6. Z menu Uruchom wybieramy Kompiluj i uruchom
7. W oknie zapisz plik wskazujemy pulpit → folder nazwisko51 → otwieramy folder nazwisko51→ klikamy zapisz

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