Tutorial

Programowanie w C++
Rozdział: Pętle

Autor: Mateusz Wojtyna

Opis ogólny

Pętle są jednymi z podstawowych zagadnień języka programowania i to nie tylko C++. Są one bardzo praktyczne w użyciu i pomocne jeżeli chcemy wykonać dany blok informacji wielokrotnie. W zastosowaniu języka C++ są stosowane 3 pętle mianowicie :
-pętla for
-pętla while
-pętla do_while

Pętla for

Najpierw opiszę pętle for ponieważ jest ona chyba najłatwiejsza w zrozumieniu i opanowaniu. Składnia pętli ma następujący wygląd:

```
for(instrukcja początkowa;warunek sterujący;instrukcja kroku)
{
 instrukcja 1;
 instrukcja 2;
 ……;
 instrukcja n;
}
```

for-słowo kluczowe dla inicjacji pętli
instrukcja początkowa-instrukcja wykonująca się tylko raz przed pierwszym obiegiem pętli zwana inicjującą
warunek sterujący -wyrażenie którego wartość logiczna jest badana przed każdym obiegiem pętli jeśli ma wartość true pętla wykona się w przeciwnym wypadku nastąpi wyjście z pętli
instrukcja kroku-instrukcja wykonywana przed każdym obiegiem pętli najczęściej modyfikuje tak zwany licznik pętli
instrukcja 1,instrukcja 2,itd..-instrukcje wykonywane w czasie obiegu pętli

Może się to wydawać trudne lecz lepiej to zobaczyć na prawdziwym przykładzie wtedy rozjaśnią się wszelkie niejasności.

Na ten przykład zróbmy program który obliczy daną potęgę X z podanej przez siebie liczby N.

Nic prostszego pierwsze oczywiście należy postępować jak w każdym innym programie zadeklarować biblioteki oraz pamięć na jakiej będziemy pracować a potem należy zaczynać pisać nasze zagadnienie oczywiście zaczynamy od deklaracji zmiennych co wyglądać będzie następująco

```
int N,Z;
```

Potem należy podać wartości zmiennych:

```
cout<<"Podaj potęge Z";
cin>>Z;
cout<<"Podaj liczbę potęgowaną N";
cin>>N;
```

Potem przechodzimy do właściwej pętli

```
int N1;
N1=N;
for(int i=1;i<Z;i++)
{N=N*N1;}
```

Teraz należy się chwila objaśnienia N1 jest to liczba którą zadeklarowałem tuż przed pętlą gdyż w potędze wartość liczby N zmienia się a liczba N1 podtrzymuję wartość początkową liczby potęgowanej aby potęgowanie miało wartość prawdziwą matematycznie potem znowu zadeklarowałem zmienną i która ma liczyć ilość już wykonanych potęgowań aż do chwili gdy pętla wykonana się do końca. Instrukcje podałem w nawiasach {} ale podczas 1 instrukcji nie jest to koniecznie gdyż program wykona pętle wtedy właśnie tylko na 1 poleceniu ale ukazałem taką postać bo zazwyczaj jest o wiele więcej instrukcji w pętli.

Teraz wystarczy wyświetlić wynik

```
cout<<N;
```

I wykonać tak jak zawsze pozostałe instrukcje programu.

Pętla while

Jest ona równie łatwą pętlą ale wygląda trochę inaczej i inaczej działa.
Składnia pętli while ma następujący wygląd :

while(wyrażenie)
{ instrukcja 1;
instrukcja 2;
……..;
instrukcja n;
}

while-jest to słowo kluczowe inicjujące pętle
wyrażenie-jest to warunek przyjmujący wartość logiczną prawda albo falsz
instrukcja1,instrukcja2,itd..-instrukcje wykonywane w pętli

Pętla while sprawdza warunek przed wykonaniem instrukcji wewnątrz siebie więc instrukcje te mogą nie być wykonane wcale

Jak widać ta pętla nie różni się dużo od poprzedniej pętli for ale tzw. zwiększanie licznika pętli musimy wykonać gdyż nie ma tu takiej instrukcji
Wyjaśnijmy to wszystko na bardzo prostym przykładzie.

Zróbmy program wykonujący mnożenie ale metodą dodawania(bo przecież mnożenie to wielokrotne dodawanie) A będzie to mnożnikiem a N liczbą mnożoną.

Postępujemy tak jak zawsze przy tworzeniu programu aż gdy dojdziemy do deklaracji zmiennych

```
int N,A;
```

Potem jak zwykle podajemy wartości zmiennych

```
cout<<"Podaj ile liczba ma być pomnożona";
cin>>A;
cout<<"Podaj liczbe";
cin>>N;
```

Przechodzimy teraz do pętli

```
int i=1,N1;
N1=N;
while(i<A)
{
 N=N+N1;
 i++;
}
```

Trzeba wyjaśnić kilka spraw w tej pętli trzeba zadeklarować zmienna pomocniczą i wcześniej oraz znowu trzeba przechować pierwsza wartość zmiennej liczbowej N aby mnożenie miało dobrą wartość końcową
Ale jest też nowość jak mówiłem tutaj trzeba „samemu" zwiększać licznik pętli więc po wykonaniu instrukcji podstawowej jest jeszcze umieszczona instrukcja zwiększenia licznika pętli czyli i++.

Teraz wystarczy wyświetlić wynik

```
cout<<N;
```

I wykonać końcowe instrukcje programu.

Pętla do…while

Jest to pętla różniąca się od poprzednich dość znacząco dlatego że warunek pętli sprawdzany jest dopiero po obiegu pętli czyli wykonaniu instrukcji umieszczonych w niej a więc pętla wykona się co najmniej jeden raz co nie oznacza że jest ona trudniejsza w zrozumieniu.
Składnia pętli while ma następujący wygląd :

```
do
{instrukcja 1;
  instrukcja 2;
 ……;
  instrukcja n;
}while(wyrażenie);
```

do-słowo kluczowe inicjujące pętle
instrukcja 1,instrukcja 2,itd..-instrukcje wykonywanie w czasie trwania pętli
while-drugie słowo kluczowe inicjujące sprawdzenie warunku pętli
wyrażenie- przyjmuje jedną z dwóch wartości logicznych:prawdę albo fałsz

Jak już mówiłem pętla **wykona się co najmniej raz** a czy wykona się więcej razy zależy już od wartości wyrażenia.

Podobnie jak w przypadku pętli while nie ma tu zwiększania licznika pętli więc musimy to zrobić sami ,pokaże przykład takiej pętli.

Mamy sprawdzić czy podana przez użytkownika programu liczba B jest większa od zera(ta pętla jest wprost idealna dla sprawdzania poprawności danych już wprowadzonych i wprowadzenia ich ponownie ale poprawnych).

Zaczynamy program jak każdy inny aż dojdziemy do deklaracji zmiennych w tym przypadku będzie to

```
int B;
```

Potem podajemy wartość w pętli aby (gdy będzie zła) łatwo było ją zmienić

```
do
{
 cout<<"Podaj liczbe ";
 cin>>B;
 if(B<0)
 cout<<"Niepoprawna wartosc podaj liczbę jeszcze raz "<<endl;
}while(B<0);
```

Oczywiście umieszczamy jeszcze komunikat aby użytkownik wiedział że zrobił coś nie tak i znowu wprowadził wartość w tym banalnym przypadku nie trzeba było używać licznika pętli bo warunek był ciągle jeden czyli ta pętla może wykonywać się bez końca jeżeli użytkownik tak zechce

Na końcu trzeba jeszcze wyprowadzić komunikat że użytkownik wprowadził dobrą liczbę

```
cout<<"Wprowadziles poprawna wartosc";
```

I wykonać końcowe polecenia.

Dziękuje za uwagę i mam nadzieję że ten poradnik pomoże udoskonalić lub nauczyć zdolności wykonywania pętli w języku programowania C++