Lekcja 3
Iteracja - wielokrotnie powtarzamy te same instrukcje

schemat – przypisanie - inkrementacja i dekrementacja – pętle zagnieżdżone - konwersja

Schemat instrukcji FOR

for (początek; koniec; iteracja) instrukcja;
for (początek; koniec; iteracja) { instrukcje;… }

Przykład

```
for (int i=1; i<=10; i=i+1)
   cout << i << endl;
```

Wypisz na ekranie 10 kolejnych liczb naturalnych

- deklarujemy **zmienną sterującą pętlą** *„i”* nadajemy jej wartość 1
- deklarujemy kiedy pętla jest kontynuowana (*i<=10*), kiedy pętla się skończy (*i>10*)
- po wykonaniu każdej instrukcji, zmienna sterująca ulega zmianie, tutaj jest zwiększana o 1
- w pętli wykonywana jest instrukcja cout, na ekranie wypisywane są liczby od 1 do 10
- nie musimy stosować instrukcji bloku {}, bo wykonujemy w pętli tylko jedną instrukcję cout

Zadanie – suma liczb

Wylicz sumę oraz iloczyn 10 kolejnych liczb całkowitych

```
int suma=0;
int iloczyn=1;
for (int i=1; i<=10; i=i+1){
  suma=suma+i;
  iloczyn=iloczyn*i;
}
cout << "SUMA=" << suma << endl;
cout << "ILOCZYN=" << iloczyn << endl;
```

- deklarujemy zmienne i nadajemy im wartości początkowe
- w pętli zwiększamy sumę i iloczyn o „i”
- ponieważ w pętli wykonujemy dwie instrukcje, dlatego blok {}
- wypisujemy sumę i iloczyn na ekranie

Inkrementacja i dekrementacja

```
for (int i=1;i<=10;i++)
  cout << i << endl;
```

Wszystkie języki programowania posługują się zapisem i=i+1 – tylko język C wprowadził „udogodnienie” polegające na skróceniu tego zapisu do i++ - dwa znaki mniej! ale za to mnóstwo zamieszania. Podobnie zamiast i=i-1 możemy zapisać i--. Jakie inne „udogodnienia” w języku C++?

```
int suma=0;
int iloczyn=1;
for (int i=1;i<=10;i++){
  suma+=i;  iloczyn*=i;
}
cout << "SUMA=" << suma << endl;
cout << "ILOCZYN=" << iloczyn << endl;
```

i=i+1	i++
i=i-1	i--
s=s+a	s+=a
s=s-a	s-=a
s=s*a	s*=a
s=s/a	s/=a

Co ciekawe, twórcy języka C przewidzieli następujące zapisy: i++ oraz ++i. Obie zwiększają wartość zmiennej „i” o jeden, ale ta pierwsza (i++) robi to po wykonaniu pętli, a druga (++i) przed wykonaniem pętli! Na wszelki wypadek stosuj zawsze instrukcję **i++**

Zadanie – Konwersja

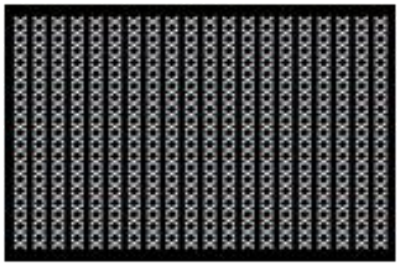
Napisz program, który wyprowadzi znaki od A do Z.

```
for (int i=65;i<=90;i++)
  cout << i << "  " << char(i) << endl;
```

char (i) albo (**char) i** zamienia liczbę z pętli na odpowiadający jej znak w kodzie ASCII. Aby sprawdzić, pod jakim numerem kryje się znak wpisany z klawiatury możesz wypróbować konwersję znaku na liczbę **int(znak)**. Aby przeprowadzić podobne (lub inne) konwersje można także korzystać z wielu gotowych funkcji.

```
char znak;
cin >> znak;
cout << int(znak) << endl;
```

Pętle zagnieżdżone
Zadanie - gwiazdki

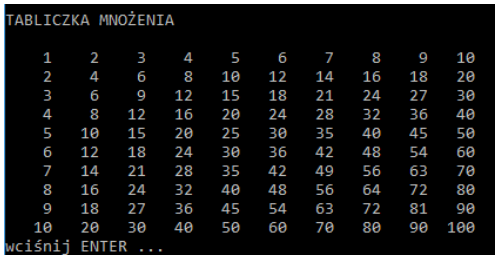


Wyświetl na ekranie prostokąt złożony z WxK gwiazdek. W i K wprowadzamy z klawiatury.

```
int W,K;
cout << "Podaj W: ";
cin >> W;
cout << "Podaj K: ";
cin >> K;
for (int i=1;i<=W;i++){
  for (int j=1;j<=K;j++){
    cout << "*";
  }
  cout << endl;
}
```

- wczytujemy liczbę wierszy i kolumn do zmiennych W i K
- wewnętrzna pętla (zmienna j) wypisuje gwiazdki w wierszu – tyle ile kolumn
- po wypisaniu gwiazdek w wierszu przechodzimy do nowego
- zewnętrza pętla (zmienna i) zlicza kolejne wiersze
- wewnętrzna pętla wykonuje tylko jedną instrukcję dlatego nie musi być klamer, ale ładniej wygląda jeśli je wstawimy i być może przydadzą się w przyszłości, gdy będziemy rozbudowywać program.

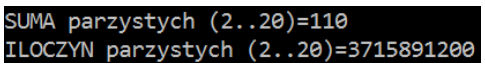
Zadanie – tabliczka mnożenia



Wyświetl na ekranie tabliczkę mnożenia w zakresie 1..100

```
for (int i=1;i<=10;i++){
  for (int j=1;j<=10;j++){
    cout.width(5);
    cout << i*j;
  }
  cout << endl;
}
```

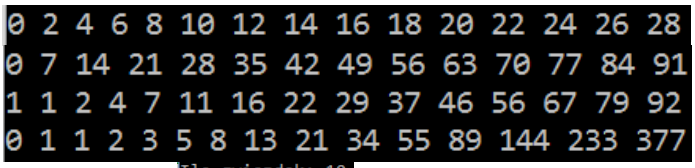
- środkowa pętla tylko jedna instrukcja, dlatego bez bloku {}
- zewnętrzna pętla dwie instrukcje: pętla i nowy wiersz, dlatego blok {}



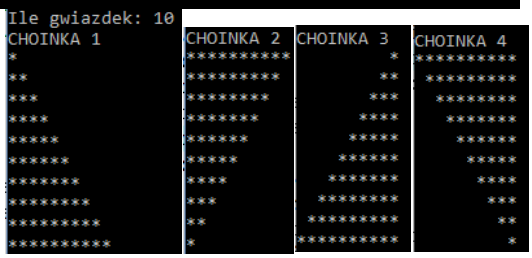
Zadania

1) *Wypisz na ekranie sumę i iloczyn liczb parzystych w zakresie 2..20*

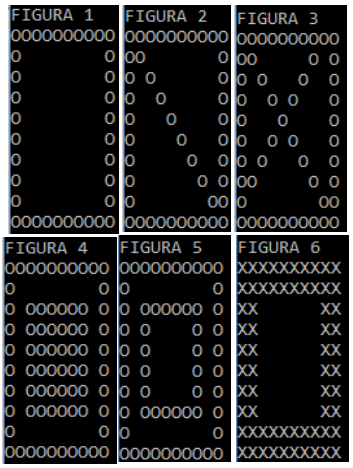
2) *Napisz programy, które wypiszą na ekranie poniższe ciągi liczbowe i 50 następnych (przy pomocy pętli FOR)*
   a) *0 2 4 6 8 10…*
   b) *0 7 14 21 28…*
   c) *1 1 2 4 7 11 16…*
   d) *0 1 1 2 3 5 8 13…*



3) *„Narysuj" na ekranie choinki złożone z gwiazdek: 1, 2, 3,… aż do N gwiazdek. Ilość gwiazdek (wiersze = kolumny) wczytujemy z klawiatury. UWAGA Choinki rysowane są zawsze od góry!*



4) *Wykorzystaj instrukcję warunkową i funkcje logiczne, i narysuj za pomocą pętli FOR poniższe obrazki. Ilość znaków „O" wpisujemy z klawiatury. UWAGA Możesz najpierw narysować prostokąt z gwiazdek, a potem za pomocą warunków logicznych rysować spacje - usuwać znaki.*
   a) *policz liczbę znaków "O" i wypisz na końcu*



5) *Napisz prosty kalkulator, który będzie działał w pętli i reagował na znaki: +-*/. Znak = pokazuje wynik obliczeń C kasuje obliczenia Q koniec działania programu.*