Podstawy programowania. Języki programowania.

Pętle

Pętla while

- Pętla while wykonuje daną instrukcję lub blok instrukcji tak długo jak długo warunek jest spełniony (ma wartość true). Ogólna postać pętli while jest następująca:

```
while (warunek)
        instrukcja;
```

- Pętla while bada prawdziwość warunku jeszcze przed wykonaniem dalszego kodu (na samym jej początku), dlatego też, jeśli warunek ma wartość false to instrukcje zawarte w tej pętli nigdy nie zostaną wykonane.

Pętla while - przykład

```
public static void main(String[] args) {
int i = 0;
while(i < 10)
{
    System.out.println(i + "Pętle w Javie");
    i++;
}

}
```

Można też tak:

```
public static void main(String[] args) {
int i = 0;
while(i++ < 10)
{
    System.out.println(i + " Pętle w Javie");
}

}
```

Pętla do while

- Pętla do while jest odmianą pętli while, a jej ogólna postać jest następująca:

```
do
        instrukcja;
while(warunek);
```

Co należy rozumieć jako: Wykonuj instrukcję dopóki warunek jest prawdziwy. Jako, iż warunek pętli sprawdzany jest na końcu, wykona się ona zawsze przynajmniej raz!

Pętla do while - przykład

```
public static void main(String[] args) {
int i = 0;
do
{
    System.out.println(i + " Pętle w Javie");
    i++;
}
while(i < 10);

}
```

ALBO...

```
public static void main(String[] args) {
int i = 0;
do
    System.out.println(i + " Pętle w Javie");
while(i++ < 9);

}
```

Pętla for

- Ogólna postać pętli for jest następująca:

```
for (wyrażenie początkowe; wyrażenie warunkowe; wyrażenie modyfikujące)
        instrukcja_do_wykonania;
```

W miejsce wyrażenia początkowego wstawiane jest wyrażenie stosowane do zainicjalizowania zmiennej służącej jako licznik wykonań pętli. Wyrażenie warunkowe określa jaki warunek musi być spełniony by przejść do kolejnego przebiegu pętli. Wyrażenie modyfikujące natomiast używane jest do modyfikacji wartości zmiennej będącej licznikiem pętli.

Pętla for - przykład

```
public static void main(String[] args) {

for (int i = 0; i < 10; i++)
    System.out.println(i + " Pętle w Javie");

}
```

Wartość zmiennej **i** widoczna tylko w bloku pętli

LUB

```
int i =0;
for (; i < 10; )
{
    System.out.println(i + " Pętle w Javie");
    i++;
}
}
```

Wartość zmiennej i widoczna poza blokiem pętli

Instrukcja break

- Instrukcja break powoduje przerwanie wykonywania pętli i opuszczenie jej bloku

Nie ma warunku zatrzymania pętli

```
public static void main(String[] args) {

for (int i = 0; ; i++ )
{
    System.out.println(i + " Pętle w Javie");
    if ( i == 9)
        break;
}

}
```

A i tak zadziała...

Instrukcja break – przykład 2

```
public static void main(String[] args) {

for (int i = 0; i < 3; i++) {
    for (int j = 0; j < 3; j++) {
        if (i == 2)
            break;
        System.out.println(i + " " + j);
    }
}

}
```

Wynik

```
0 0
0 1
0 2
1 0
1 1
1 2
```

Instrukcja continue

- Instrukcja continue powoduje przejście do kolejnej iteracji danej pętli (chyba, że była to jej ostatnia iteracja).

```
public static void main(String[] args) {

for (int i = 0; i < 5; i++) {
    if (i == 3)
        continue;
    System.out.println(i);
}

}
```

Wynik

```
0
1
2
4
```

Etykietowane break i continue

- Instrukcje break i continue mają również postać w połączeniu z etykietami.
- Etykieta to wyznaczone miejsce w kodzie programu. Etykietę definiuje się przez podanie jej nazwy zakończonej znakiem dwukropka
- **break etykieta** powoduje przerwanie wszystkich pętli znajdujących się za etykietą (innymi słowy przerwij działanie pętli i idź do etykietaı).
- **continue etykieta** powoduje przerwanie bieżącej iteracji pętli przejście do miejsca oznaczonego etykietą. Następnie następuje ponowne wejście do pętli znajdującej się tuż za etykietą!

Etykietowane break - przykład

```
etykieta1:
for (int i = 0; i < 3; i++) {
    for (int j = 0; j < 3; j++) {
        if (i == 1)
            break etykieta1;
        System.out.println(i + " " + j);
    }
}
```

Wynik

```
0 0
0 1
0 2
```

Widać różnicę?

```
for (int i = 0; i < 3; i++) {
    for (int j = 0; j < 3; j++) {
        if (i == 1)
            break;
        System.out.println(i + " " + j);
    }
}
```

Wynik

```
0 0
0 1
0 2
2 0
2 1
2 2
```

Ćwiczenia 1

1. Wykorzystując pętle for napisz program, który wyświetli parzyste liczby całkowite z zakresu od 0 do 20.
2. Wykorzystując pętle for napisz program, który wyświetli liczby całkowite od 1 do 30 podzielne przez 3.
3. Napisz program, który wyświetli na ekranie liczby od 1 do 20 i wypisze przy każdej, czy jest ona parzysta czy nieparzysta
   - Wykorzystując pętlę for.
   - Wykorzystując pętlę while.
   - Wykorzystując pętlę do while.

Ćwiczenia 2

4. Napisz program, który wyświetli na ekranie liczby z zakresu od 1 do 100 podzielne przez 4, ale niepodzielne przez 8 i niepodzielne przez 10. Wykorzystaj w tym celu instrukcję continue.
5. Napisz program, określający ile lat trzeba oszczędzać w banku na 5% lokacie, aby przy zarobkach rzędu 12000 zł rocznie netto mieć na koncie sumę co najmniej 200000 zł. Załóż, że od odsetek ani dochodu nie jest pobierany żaden podatek.

Ćwiczenia 3

7. Napisz program wyliczający największy wspólny dzielnik dwóch liczb całkowitych podanych przez użytkownika algorytmem Euklidesa.
8. Napisz program wyznaczający silnie podanej przez użytkownika liczby.
9. Napisz program kalkulator, który będzie realizował następujące operacje:
   - Dodawanie dwóch liczb
   - Odejmowanie dwóch liczb
   - Dzielenie dwóch liczb
   - Mnożenie dwóch liczb
   - Wyznaczanie pierwiastka kwadratowego z liczby
   - Wyznaczanie procent z liczby.
   - Wyznaczanie reszty z dzielenia dwóch liczb.
   - Wyznaczanie dowolnej potęgi danej liczby.

   Kalkulator powinien umożliwiać wybór operacji do momentu podania przez użytkownika znaku „k” oznaczającego koniec działania programu.

Ćwiczenia 4

10. Napisz program, który wyznacza największą i najmniejszą wprowadzoną przez użytkownika liczbę. Zakończenie wprowadzania liczb określa znak ‘k’.
11. Napisz program wypisujący wszystkie potęgi liczby 2 aż do 2^n (n – podawane na wejściu przez użytkownika).
12. Wczytujący pojedyncze znaki (aż do momentu wystąpienia znaku ‘z’) i wyświetlający je. Po wpisaniu znaku litery ‘x', następny wczytany znak nie zostaje wyświetlony.