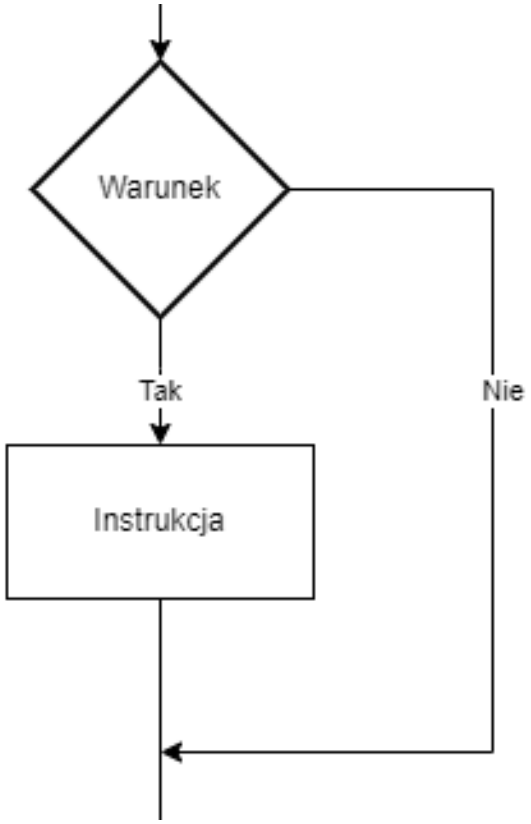
Programowanie 1

Sławomir Pluciński (splucinski@pjwstk.edu.pl)

Instrukcje sterujące

- W instrukcjach sterujących podejmowane są decyzje o wykonaniu tych lub innych instrukcji programu.
- Decyzje te podejmowane są w zależności od spełnienia lub niespełnienia określonego warunku, inaczej mówiąc od prawdziwości lub fałszywości jakiegoś wyrażenia.
- W języku C++ zdefiniowano specjalny typ określający zmienne logiczne czyli takie, które przyjmują wartości: - prawda - fałsz. Jest to typ **bool**.

Instrukcja warunkowa If



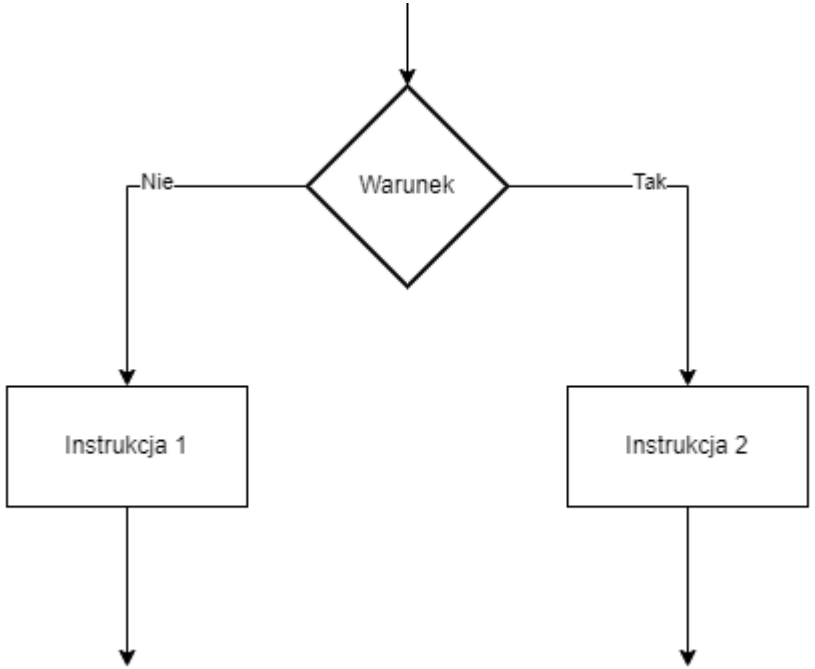
Instrukcja pozwalająca na sterownie wykonania poszczególnych instrukcji w obrębie programu w zależności od spełnienia zaproponowanego warunku.

Jeżeli warunek jest spełniony to instrukcja jest wykonywana, w przeciwnym razie wykonanie instrukcji jest pomijane.

Zapis w C++

```
if(warunek) {
    instrukcja;
}
```

Instrukcja warunkowa If Else

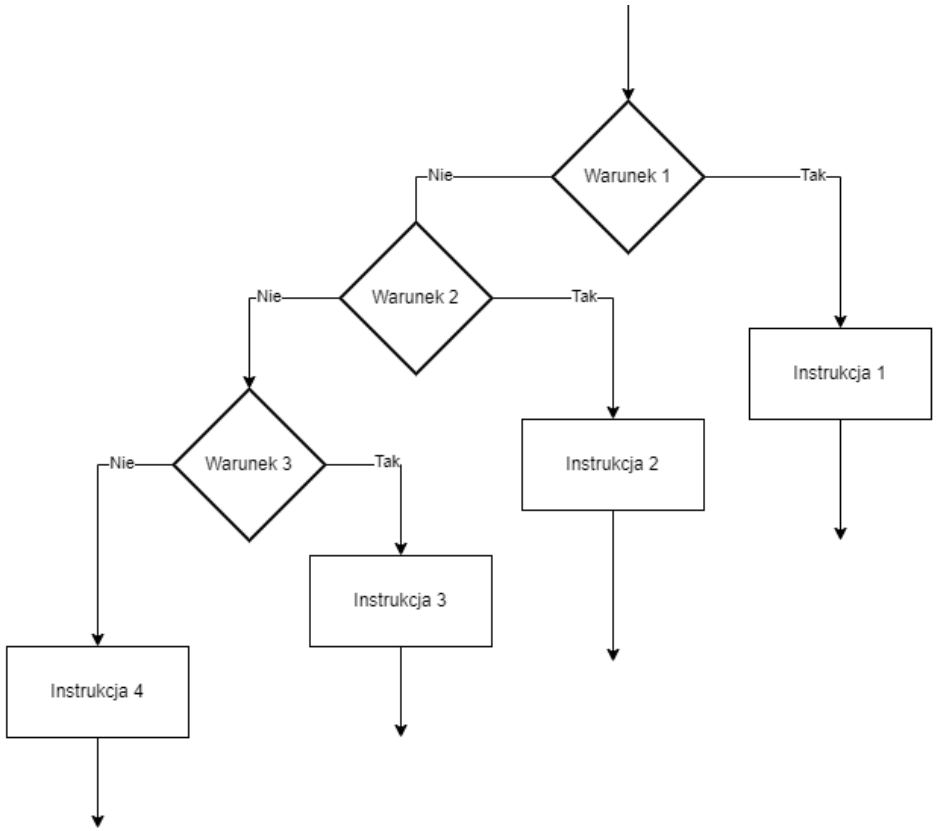


Instrukcja sterujący z alternatywą. Każdy przebieg przez warunek będzie skutkował realizacją części bloku, gdzie warunek jest spełniony lub w przypadku, gdy nie jest spełniony część alternatywy zostanie wykonana.

Zapis w C++

```
if(warunek) {
    instrukcja2;
}
else {
    instrukcja1;
}
```

Instrukcja warunkowa Else if

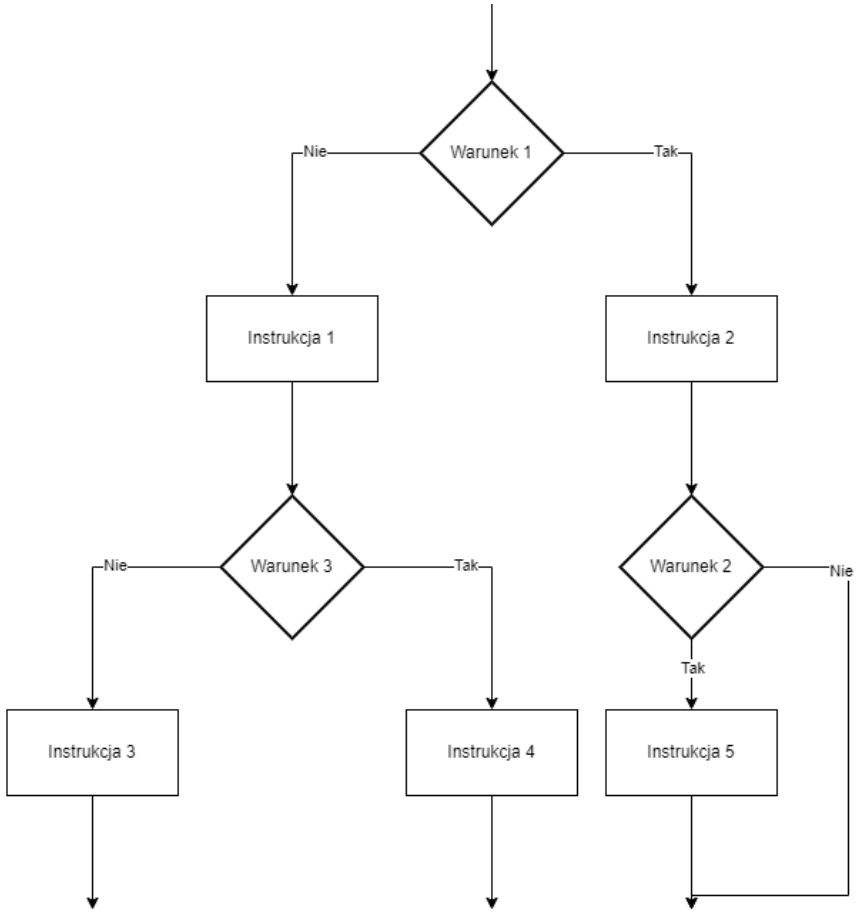


Instrukcja sterujący z alternatywą warunkową. Jeżeli warunek nie jest spełniony to program przechodzi do kolejnej instrukcji, gdzie sprawdzany jest warunek ponownie. Jeżeli blok instrukcji jest zakończony warunkiem else to tam program zostanie skierowany w sytuacji braku spełnienia jakiegokolwiek warunku else if.

Zapis w C++

```
if(warunek1) {
    instrulcja1;
}
else if(warunek2){
    instrulcja2;
}
else if(warunek3){
    instrulcja3;
}
else{
    instrulcja4;
}
```

Instrukcje warunkowe zagnieżdżone



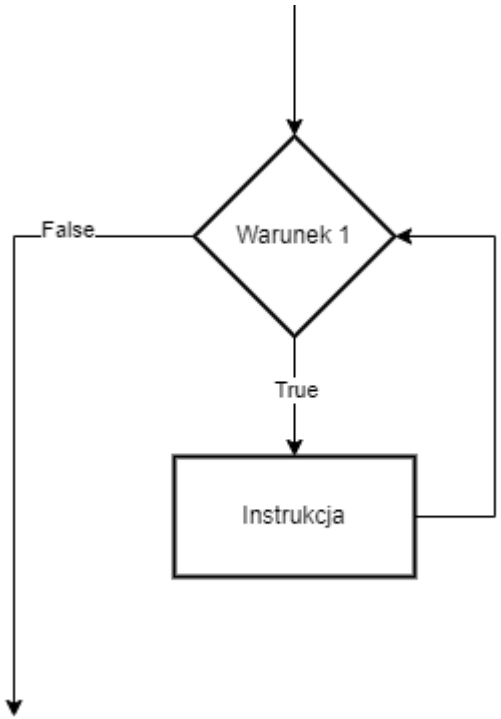
Instrukcje sterująca można zagnieżdżać, dzięki czemu tworzone są bardziej zawiłe konstrukcje pozwalające na sterowanie wykonywaniem instrukcji zależnych od siebie.

Zapis w C++

```
if(warunek1) {
    instrukcja2;

    if(warunek2){
        instrukcja5;
    }
}
else {
    instrukcja1;
    if(warunek3){
        instrukcja4;
    }
    else {
        instrukcja
    }
}
```

Instrukcja while

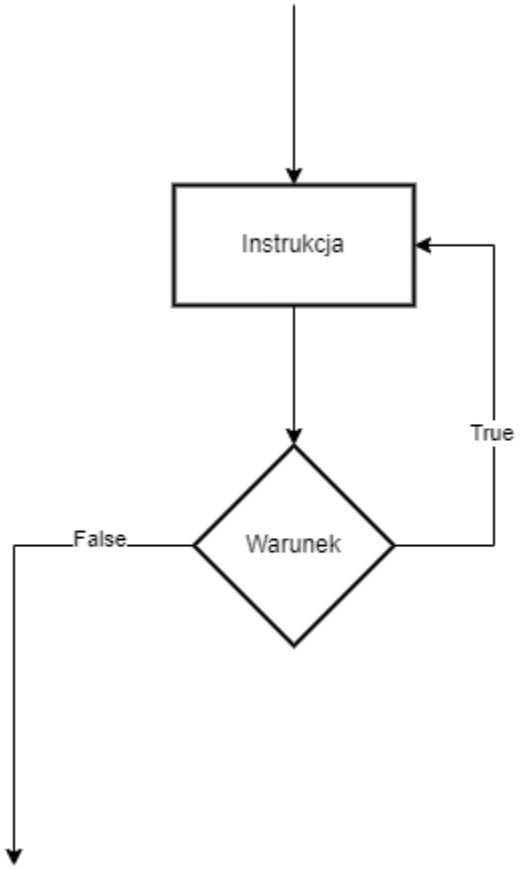


Sprawdzany jest warunek i do czasu, kiedy jest on spełniony realizowana jest zadana sekwencja kroków. Zmiana zwracanej wartości przez warunek kończy sekwencje.
Należy pamiętać, że wartość warunku jest określana przed wykonaniem instrukcji, a instrukcja może nie zostać wykonana ani razu.

Zapis w C++

```
while(warunek1){
    instrukcja;
}
```

Instrukcja do while

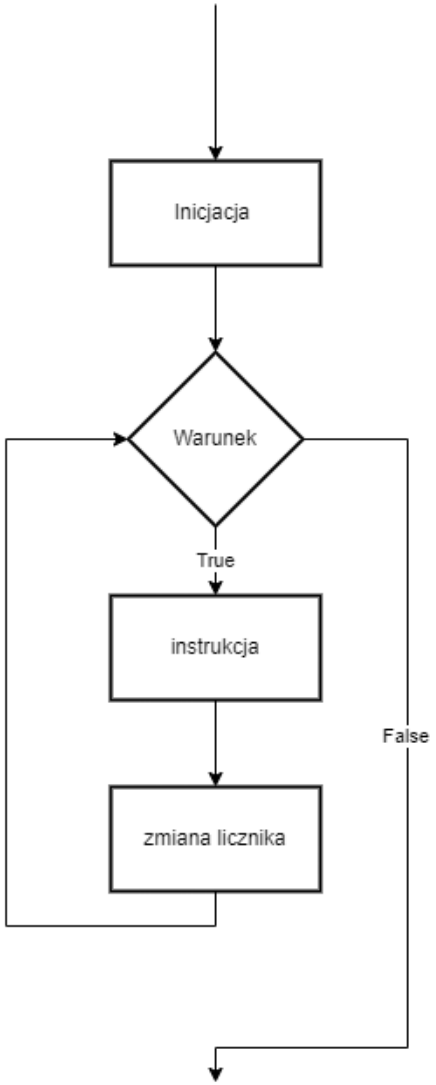


Sprawdzenie warunku poprzedzone jest realizacją instrukcji. Po wykonaniu operacji w elemencie ‚do' sprawdzany jest warunek i do póki zwraca on wartość ‚true' to wykonywana jest instrukcja.
Należy zwrócić uwagę, że w przypadku tej konstrukcji, instrukcja zawsze zostanie wykonana chociaż raz.

Zapis w C++

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

Instrukcja for



Wykonywanie instrukcji zaczyna się od wykonania raz pierwszego elementu instrukcji. Następnie sprawdzany jest warunek instrukcji i jeżeli jest ‚prawdą' to wykonywane jest ciało instrukcji for. Przed przejściem do ponownego sprawdzenia warunku, realizowany jest ostatni element instrukcji for, czyli zmiana licznika. Pętla jest realizowana do czasu zwracania ‚prawdy' w ramach sprawdzanego warunku.

Zapis w C++

```
for(zmienna; warunek; zmiana_licznika){
    instrukcja;
}

for(int i=0; i<10; i++){
    instrukcja;
}
```

Instrukcja for - dodatkowe uwagi

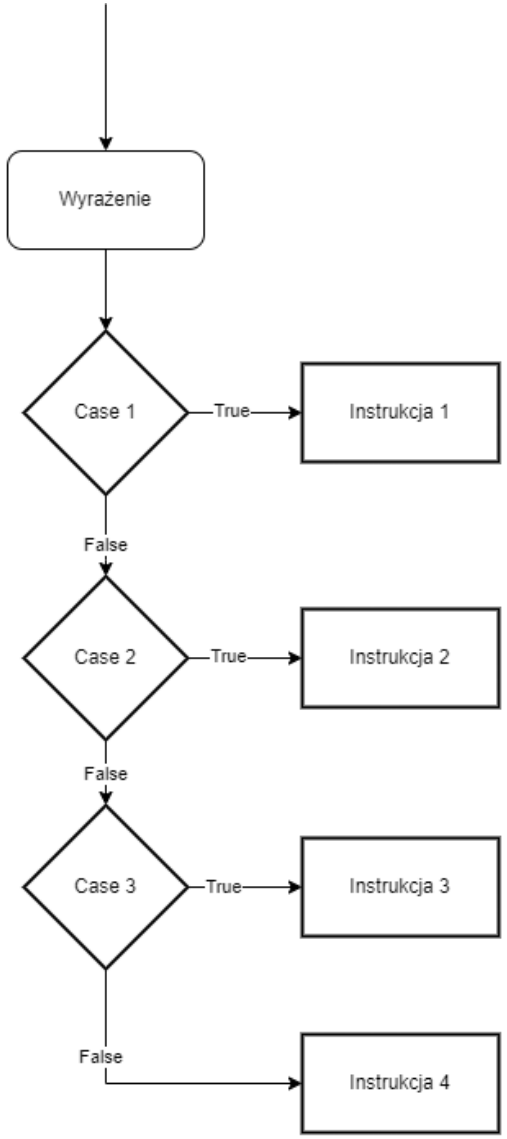
- Instrukcji inicjujących może być kilka;
  Wówczas muszą być one oddzielone przecinkami; podobnie jest w przypadku instrukcji kroku zmiana licznika;
- Wyszczególnione elementy: ini, warunek, zmiana licznika nie muszą wystąpić;
- Dowolny z nich można opuścić, zachowując jednak średnik oddzielający go od sąsiada;
- Opuszczenie wyrażenia warunkowego jest traktowane tak, jakby stało tam wyrażenie zawsze prawdziwe.

```
for(;;){

}
```

```
for(int i=0,j=10;i+j<100;i+=2,j--){
    instrukcja;
}
```

Instrukcja switch



- Obliczane jest wyrażenie umieszczone w nawiasach po słowie switch;
- jeśli jego wartość odpowiada którejś z wartości podanej w jednej z etykiet case, wówczas wykonywane są instrukcje począwszy od tej etykiety. Wykonywanie ich kończy się po napotkaniu instrukcji break. Działanie instrukcji switch zostaje wówczas zakończone;
- jeśli wartość wyrażenia nie zgadza się z żadną z wartości podanych w etykietach case, wówczas wykonywane są instrukcje umieszczone po etykiecie default.
- Etykieta default może być umieszczona w dowolnym miejscu instrukcji switch, nawet na samym jej początku. Co więcej, etykiety default może nie być wcale. Wówczas, jeśli w zbiorze etykiet case nie ma żadnej etykiety równej wartości wyrażenia, instrukcja switch nie będzie wykonana. I
- Instrukcje występujące po etykiecie case nie muszą kończyć się instrukcją break. Jeśli jej nie umieścimy, to będą wykonywane instrukcje umieszczone pod następną etykietą case.

Zapis w C++

```
switch(wyrazenie){
    case Case1: instrukcja1; break;
    case Case2: instrukcja2; break;
    case Case3: instrukcja3; break;
    default: instrukcja4; break;
}
```

Instrukcja break

- instrukcja break jest używana również w instrukcjach pętli for, while, do…while;
- instrukcja break powoduje natychmiastowe przerwanie wykonywania tych pętli;
- jeśli mamy do czynienia z pętlami zagnieżdżonymi, to instrukcja break powoduje przerwanie tylko tej pętli, w której została bezpośrednio użyta;
- jest to więc przerwanie z wyjściem o jeden poziom wyżej.

Instrukcja continue

- instrukcja continue przydaje się wewnątrz pętli for, while, do...while;
- powoduje ona zaniechanie realizacji instrukcji będących treścią pętli, jednak (w odróżnieniu od instrukcji break) sama pętla nie zostaje przerwana;
- instrukcja continue przerywa tylko ten obieg pętli i zaczyna następny, kontynuując pracę pętli.