Programowanie I
Wykład 2

dr inż. Rafał Brociek

Wydział Matematyki Stosowanej
Politechnika Śląska

11.10.2021

Instrukcja warunkowa

Warunek jest dowolnym wyrażeniem posiadającym wartość logiczną (`true` albo `false`) np. `x < 5`, `x == 1`, `((x < 2) && (x != 0))`.
Wartość zero rozumiana jest jako fałsz. Wartość inna niż zero rozumiana jest jako prawda. Warunek może być bardzo skomplikowanym wyrażeniem, które po obliczeniu zwróci wartość liczbową.

Schemat instrukcji warunkowej `if`

```
1 if(warunek)
2     instrukcja;
3
4 if(warunek)
5 {
6     blok poleceń
7 }
```

Instrukcja warunkowa

Schemat instrukcji warunkowej if

```
1  if(warunek)
2      gdy warunek jest prawdziwy;
3  else
4      gdy warunek jest fałszywy;
5
6  if(warunek)
7  {
8      blok poleceń w przypadku
9      prawdziwego warunku
10 }
11 else
12 {
13     blok instrukcji w przypadku
14     fałszywego warunku
15 }
```

Instrukcja warunkowa - przykład

```
1  bool czyKoniecGry = true;
2  if(czyKoniecGry)
3     cout << "Koniec gry! " << endl;
4  else
5     cout << "Gramy dalej." << endl;
6
7  int licznik = 9;
8  if(licznik != 0)
9  {
10    licznik--;
11    cout << "licznik zmienil wartosc";
12 }
13 else
14 {
15    // instrukcje w przypadku
16    // gdy licznik osiągnie 0
17 }
```

Instrukcja warunkowa

Schemat instrukcji warunkowej `if`

```
1  if(warunek1){
2      gdy warunek = true
3  }
4  else if(warunek2){
5      gdy waunek1 = false
6      i warunek2 = true
7  }
8  else if(warunek3){
9      gdy waunek1 = false
10     i warunek2 = false
11     i warunek3 = true
12 }
13 else{
14     gdy waunek1 = false
15     i warunek2 = false
16     i warunek3 = false
17 }
```

Instrukcja warunkowa - przykład

```
1  int miesiac = 1, pora_roku = 0;
2  cout << "Podaj numer miesiaca [1-12]: ";
3  cin >> miesiac;
4
5  if((miesiac >= 3) && (miesiac <= 5)){
6     pora_roku = 1;
7     cout << "Wiosna";
8  }
9  else if((miesiac >= 6) && (miesiac <= 8)){
10    pora_roku = 2;
11    cout << "Lato";
12 }
13 else if((miesiac >= 9) && (miesiac <= 11)){
14    pora_roku = 3;
15    cout << "Jesien";
16 }
17 else{
18    pora_roku = 4;
19    cout << "Zima";
20 }
```

Instrukcja warunkowa - przykład

```
1 // UWAGA!!!
2 int liczba = -5;
3 if(liczba = 0)
4     cout << "Rowna zero" << endl;
5 else if(liczba < 0)
6     cout << "Ujemna" << endl;
7 else
8     cout << "Dodatnia" << endl;
```

Co zostanie wypisane na ekranie?

Instrukcja warunkowa - przykład

```
1 // UWAGA!!!
2 int liczba = -5;
3 if(liczba = 0)
4     cout << "Rowna zero" << endl;
5 else if(liczba < 0)
6     cout << "Ujemna" << endl;
7 else
8     cout << "Dodatnia" << endl;
```

Co zostanie wypisane na ekranie?

Wynik: Dodatnia

Instrukcja switch

Schemat instrukcji switch

```
1  switch(wyrazenie)
2  {
3      case n1:
4          instrukcje, gdy wyrazenie
5          jest rowne n1
6          break;
7      case n2:
8          instrukcje, gdy wyrazenie
9          jest rowne n2
10         break;
11     ...
12     default:
13         instrukcje, gdy wyrazenie
14         jest różne od wszystkich poprzednich
15         break;
16 }
```

Instrukcja switch - przykład

Wyrażenie wybierające powinno być typu całkowitoliczbowego (lub elementem typu wyliczeniowego) i porównywane tylko ze stałymi całkowitoliczbowymi.

```
1  cout << "Podaj cyfre od 0 do 3" << endl;
2  char z = 0;
3  cin >> z;
4  switch (z)
5  {
6  case 48: cout << "zero" << endl;
7     break;
8  case 49: cout << "jeden" << endl;
9     break;
10 case 50: cout << "dwa" << endl;
11    break;
12 case 51: cout << "trzy" << endl;
13    break;
14 default:
15    cout << "Nie podales cyfry od 0 do 3" << endl;
16    break;
17 }
```

Operator warunkowy ?

Operator warunkowy ?

warunek ? instruckjaP : instrukcjaF

```
1  // przykład 1
2  double x = 0;
3  cin >> x;
4  cout << "Liczba " << x << " jest ";
5  cout << (x > 0 ? "dodatnia" : "niedoadatnia") << endl;
6
7  // przykład 2
8  double x = 0, y = 0;
9  cin >> x >> y;
10 cout << x << " jest ";
11 cout << (x > y ? "wieksze" : (x < y ? "mniejszy" : "rowny"));
12 cout << " od " << y << endl;
```

Pętle

Pętla – jedna z podstawowych konstrukcji programistycznych umożliwiająca cykliczne wykonywanie instrukcji (bloku instrukcji).
Wyróżniamy pętle:

- **iteracyjne** – wykonuje się ustaloną z góry liczbę razy (odpowiada za to licznik pętli),
- **repetycyjne** (warunkowe) – wykonują się do momentu zajścia pewnych warunków,
- nieskończone – nas nie interesujące.

Pętla do-while

Schemat pętli do-while

```
1 do
2     instrukcja;
3 while(warunek)
4
5 do
6 {
7     blok instrukcji
8 }
9 while(warunek)
```

Najpierw wykonywana jest instrukcja (blok instrukcji), a następnie sprawdzany jest warunek.

Pętla do-while - przykład

```
1  // przykład 1
2  int i = 0;
3  int max_iter = 10;
4  do
5      cout << i << " ";
6  while (i++ < max_iter);
7
8  // przykład 2
9  cout << endl;
10 do
11 {
12     cout << i << " ";
13     i -= 2;
14 } while (i > 0);
```

Pętla do-while - przykład

```
1  // przykład 1
2  int i = 0;
3  int max_iter = 10;
4  do
5      cout << i << " ";
6  while (i++ < max_iter);
7
8  // przykład 2
9  cout << endl;
10 do
11 {
12     cout << i << " ";
13     i -= 2;
14 } while (i > 0);
```

Wynik:

0 1 2 3 4 5 6 7 8 9 10

Pętla do-while - przykład

```
1  // przykład 1
2  int i = 0;
3  int max_iter = 10;
4  do
5     cout << i << " ";
6  while (i++ < max_iter);
7
8  // przykład 2
9  cout << endl;
10 do
11 {
12    cout << i << " ";
13    i -= 2;
14 } while (i > 0);
```

Wynik:

0 1 2 3 4 5 6 7 8 9 10

11 9 7 5 3 1

Pętla while

Schemat pętli `while`

```
1 while(warunek)
2     instrukcja;
3
4 while(warunek)
5 {
6     blok instrukcji
7 }
```

Najpierw sprawdzany jest warunek, a następnie wykonywana jest instrukcja (blok instrukcji).

Pętla while - przykład

```
1  int i = 0;
2  int max_iter = 10;
3  while (i++ < max_iter)
4      cout << i << " ";
5  cout << endl;
6  while (i > 0)
7  {
8      cout << i << " ";
9      i -= 2;
10 }
```

Pętla while - przykład

```
1  int i = 0;
2  int max_iter = 10;
3  while (i++ < max_iter)
4      cout << i << " ";
5  cout << endl;
6  while (i > 0)
7  {
8      cout << i << " ";
9      i -= 2;
10 }
```

Wynik:

1 2 3 4 5 6 7 8 9 10

11 9 7 5 3 1

Pętla while - przykład

```
1 char litera = 'a';
2 cout << "Wpisuj kolejne litery" << endl;
3 cout << "litera 'k' przerywa petle" << endl;
4 while(litera != 'k')
5 {
6     cin >> litera;
7     cout << litera << endl;
8 }
9 cout << endl << "koniec" << endl;
```

Pętla for

Schemat pętli for

```
1 for(instr_inicjalizujaca ; warunek ; instr_kroku)
2    treść pętli;
3
4 for(instr_inicjalizujaca ; warunek ; instr_kroku)
5 {
6    treść pętli
7 }
```

```
1 // przykład silnia
2 int silnia = 1;
3 int n = 5;
4 for(int i=2; i<=n; i++)
5    silnia *= i;
6 cout << n << "! = " << silnia << endl;
```

Instrukcja break

Instrukcja **break** powoduje natychmiastowe przerwanie wykonywania się pętli.

```
1  for(int i=0; i<=4; i++)
2  {
3      for(int j=0; j<=4; j++)
4      {
5          if(i == j)
6              break;
7          cout << i << "*" << j;
8          cout << " = " << i*j << "   ";
9      }
10     cout << endl;
11 }
```

Instrukcja break

```
#include <iostream>
#include <thread>
using namespace std;

int main()
{
    int licznik = 10;
    cout << "Rozpoczeto odliczanie:" << endl;
    while (true)
    {
        if (licznik <= 0)
            break;
        this_thread::sleep_for(1s);
        cout << licznik-- << endl;
    }
    cout << "Koniec " << endl;
}
```

Instrukcja continue

Instrukcja **continue** powoduje natychmiastowe przerwanie bieżącego obiegu pętli, jednak sama pętla nie zostaje przerwana.

```
1  for(int i=0; i<=4; i++)
2  {
3      for(int j=0; j<=4; j++)
4      {
5          if(i <= j)
6              continue;
7          cout << i << "*" << j;
8          cout << " = " << i*j << "   ";
9      }
10     cout << endl;
11 }
```

Instrukcja break/continue

```
1  char znak;
2  cout << "Cyfry beda pomijane." << endl;
3  cout << "W celu zakonczenia petli wpisz litere 'k'.";
4  cout << endl << endl;
5  while (true)
6  {
7      cout << "Wpisz znak:  ";
8      cin >> znak;
9      if(znak == 'k')
10         break;
11     if(znak >= '0' && znak <= '9')
12         continue;
13     cout << "Wpisales: " << znak << endl;
14 }
15 cout << endl << endl;
16 cout << "Koniec wpisywania" << endl;
```

Podsumowanie cz. I

- instrukcja warunkowa `if(warunek)` służy do testowania `warunku` i w zależności od prawdziwości warunku wykonania odpowiednich dalszych instrukcji,
- warunek jest wyrażeniem posiadającym wartość logiczną (`true` albo `false`),
- możemy tworzyć złożone instrukcje warunkowe `if(...) else if(...) else`,
- instrukcją `switch(wyrazenie)` testujemy `wyrazenie` i na tej podstawie wybrany zostaje odpowiedni przypadek (`case`),
- wyrażenie wybierające w instrukcji `switch` powinno być typu całkowitoliczbowego i być porównywane ze stałymi całkowitymi,
- operator warunkowy `?` ma postać:
  `warunek ? ins_prawda : ins_falsz`

Podsumowanie cz. II

- rozróżniamy trzy rodzaje pętli: `for`, `do-while`, `while`,
- zmienne tworzone wewnątrz pętli mają zakres istnienia ograniczony do pętli, w której zostały utworzone,
- instrukcja `break` powoduje natychmiastowe przerwanie wykonywania pętli,
- instrukcja `continue` powoduje natychmiastowe przerwanie bieżącego obiegu pętli. Sama pętla nie zostaje przerwana.

Zadanie

Zadanie 1. Napisz program, który dla zadanego przez użytkownika argumentu x oblicza wartość funkcji:

$$f(x) = \begin{cases} x + 1 & \text{gdy } x < 0 \\ sin(x) & \text{gdy } 0 \leq x < 5 \\ x^2 - x + 1 & \text{gdy } x \geq 5 \end{cases}$$

Zadanie

Zadanie 2. Napisz program, który oblicza sumę kolejnych liczb całkowitych z przedziału $[a, b]$, gdzie liczby $a, b \in \mathbb{Z}$ $(a < b)$ zadaje użytkownik. Wypisz tę sumę wraz z liczbami na nią się składającymi.

```
Podaj a: -2
Podaj b: 5
(-2) + (-1) + 0 + 1 + 2 + 3 + 4 + 5 = 12
Press any key to continue . . .
```

Dziękuję za uwagę