INFORMATYKA

C

WYDZIAŁ ELEKTROTECHNIKI I INFORMATYKI
ELEKTROTECHNIKA
DR INŻ. MICHAŁ ŁANCZONT

VI

Zakres wykładu

Powtarzanie kodu

1. *Skok w kodzie – `goto`*
2. *Pętla iteracyjna – `for`*
3. *Instrukcje przerwania pętli*
   - `break`
   - `continue`
4. *Pętle warunkowe*
   - `while`
   - `do while`

Powtarzanie kodu

- Możliwość powtórzenia określonego fragmentu kodu zadaną liczbę razy jest równie istotne jak podejmowanie decyzji przez program
- We wszystkich współczesnych językach programowania dostępne są konstrukcje umożliwiające wielokrotne wykonanie określonego kodu

Instrukcja skoku

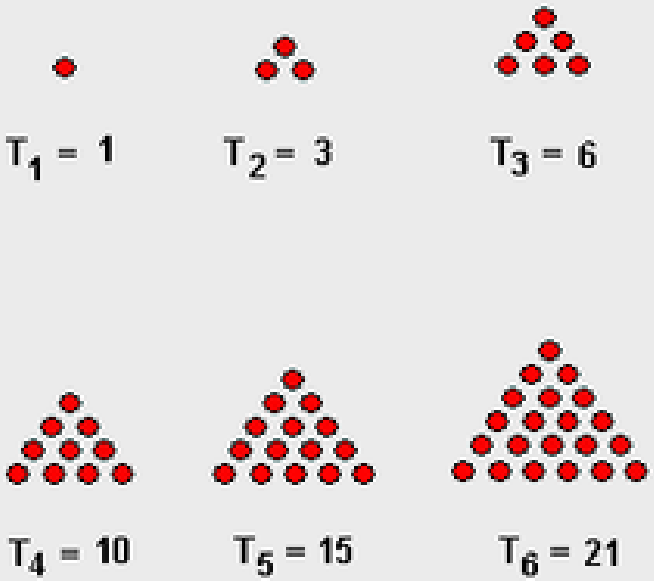
- Język C odziedziczył po językach z których się wywodzi instrukcję skoku
- Instrukcja ta działa analogicznie do skoku realizowanego w instrukcji wyboru `switch..case`
- Instrukcja `goto` przeskakuje z wykonywaniem kodu do linii zawierającej etykietę wskazaną przez instrukcję `goto`
- Skok możliwy jest zarówno w górę jak i w dół kodu
- Instrukcja `goto` zazwyczaj współpracuje z instrukcją warunkową `if` która steruje liczbą powtórzeń

Instrukcja skoku

- Podstawowa składnia

```
    kod;
etykieta_1:
    kod;
    if(warunek) goto etykieta_2;
    kod;
etykieta_2:
    kod;
    if(warunek) goto etykieta_1;
```

Przykład



$$T_n = \frac{n(n+1)}{2}$$

- Program wyświetla ciąg kolejnych liczb trójkątnych dla określonego szeregu liczb całkowitych
- Użytkownik podaje liczbę całkowitą, a program generuje ciąg liczb trójkątnych

Przykład

```c
#include <stdio.h>

int main()
{
    printf("Program generuje szereg kolejnych liczb trojkatnych.\n");
    printf("Podaj ile liczb trojkatnych wygenerowac, n=");
```

C:\Windows\system32\cmd.exe

```
Program generuje szereg kolejnych liczb trojkatnych.
Podaj ile liczb trojkatnych wygenerowac, n=10
T(1)     : 1
T(2)     : 3
T(3)     : 6
T(4)     : 10
T(5)     : 15
T(6)     : 21
T(7)     : 28
T(8)     : 36
T(9)     : 45
T(10)    : 55

Press any key to continue . . .
```

Pętla iteracyjna

1. W językach programowania dostępne są dwa typy pętli:
   - Iteracyjna
   - Warunkowa
2. Pętle iteracyjne wykonują określony kod zadeklarowaną liczbę razy.
3. Pętle iteracyjne sterowane są zmienną iteracyjną definiowaną w deklaracji pętli
4. Przy każdym powtórzeni pętli parametr ten jest inkrementowany lub dekrementowany
5. Koniec iteracji pętli określany jest jednym z parametrów deklaracyjnych

Pętla iteracyjna

- W języku C dostępna jest pętla iteracyjna `for`

`for(deklaracja;warunek;inkrementacja)`

- Pętla for sterowana jest przez zmienną typu `int`
- Definicja pętli `for` określona jest przez trzy parametry:

a) **Deklaracja** – inicjacja parametru sterującego
b) **Warunek** – warunek powtórzenia pętli
c) **Inkrementacja** – sposób zmiany parametru sterującego przy każdej iteracji pętli

- Parametry deklaracyjne pętli `for` nie są obowiązkowe

Pętla iteracyjna - deklaracja

- Ustawienie wartości początkowej zmiennej sterującej

```
for(i=0;K;I)
```

- Inicjacja zmiennej sterującej

```
for(int i=0;K;I)
```

Zmienna sterująca jest zmienną lokalną kodu pętli

- Wykorzystanie zmiennej zadeklarowanej w danej funkcji bez zmiany jej wartości

```
for(;K;I)
```

Pętla iteracyjna - deklaracja

- W obszarze deklaracyjnym pętli `for` można zainicjować bądź zadeklarować więcej niż jedną zmienną sterującą

```
for(int i=0,int j=0;K;I)
```

- Analogicznie można także określić warunek powtórzenia i inkrementację dla każdego z zainicjowanych parametrów pętli

Pętla iteracyjna - warunek

- Warunek – drugi parametr pętli `for` – określa dla jakiej wartości zmiennych sterujących kod pętli może być powtórzony
- Brak warunku oznacza że pętla będzie powtarzana w nieskończoność
- Warunek można określić dla dowolnego parametru (zmiennej)

Pętla iteracyjna - warunek

- Zazwyczaj w deklaracji określa się początkową wartość iteratora pętli, a w warunku końcową
- Warunek po zazwyczaj postać warunku logicznego, jeżeli warunek zwraca wartość TRUE pętla będzie wykonana, jeżeli zwróci FALSE nie zostanie wykonana
- Może się zdarzyć że pętla nie zostanie nawet raz wykonana, zazwyczaj jeżeli warunek jest powiązany z parametrami zewnętrznymi

Pętla iteracyjna - warunek

```
for(D;i<10;I)

for(D;i<j;I)

for(D;test(i);I)

for(D;i<10 && j<30;I)

for(D;;I)
```

Pętla iteracyjna - inkrementacja

- Trzeci parametr określa zmianę iteratora pętli
- Zmiana może być dowolnie określona
- Podobnie jak pozostałe parametry inkrementacja też nie jest parametrem wymaganym
- Inkrementacja o jeden – `i++`
- Dekrementacja o jeden – `i--`
- Inkrementacji może podlegać dowolna liczba parametrów pętli

Pętla iteracyjna - inkrementacja

```
for(D;W;i++)

for(D;W;)

for(D;W;i++,j--)

for(D;W;i=i*i)

for(D;W;i=i+j)
```

Przykład

```c
#include <stdio.h>

int main()
{
    printf("Test petli teracyjnej dwoch zmiennych.\n");
    printf("Podaj ograniczenie zmiennej i=");
    int X;
    scanf("%i",&X);
    for(int i=0, j=0;i<=X || j<=5*X;i++,j=j+i)
        printf("i=%i\tj=%i\n",i,j);
}
```

```
C:\Windows\system32\cmd.exe
Test petli teracyjnej dwoch zmiennych.
Podaj ograniczenie zmiennej i=5
i=0     j=0
i=1     j=1
i=2     j=3
i=3     j=6
i=4     j=10
i=5     j=15
i=6     j=21

Press any key to continue . . .
```

```
Y=5

Press any key to continue . . .
```

Przerwanie pętli

- Może się zdarzyć że w pewnych sytuacjach w trakcie wykonywania iteracji pętli zajdzie konieczność przerwania lub pominięcia iteracji
- Język C dostarcza dwóch funkcji przerwania wykonywania kodu w aktywnym bloku {}
- Funkcja `break` znana z instrukcji wyboru powoduje wyjście z pętli i przejście do wykonywania pierwszego wiersza po kodzie pętli
- Funkcja `continue` przerywa działanie aktualnej iteracji pętli w wymusza rozpoczęcie następnej

Przerwanie pętli

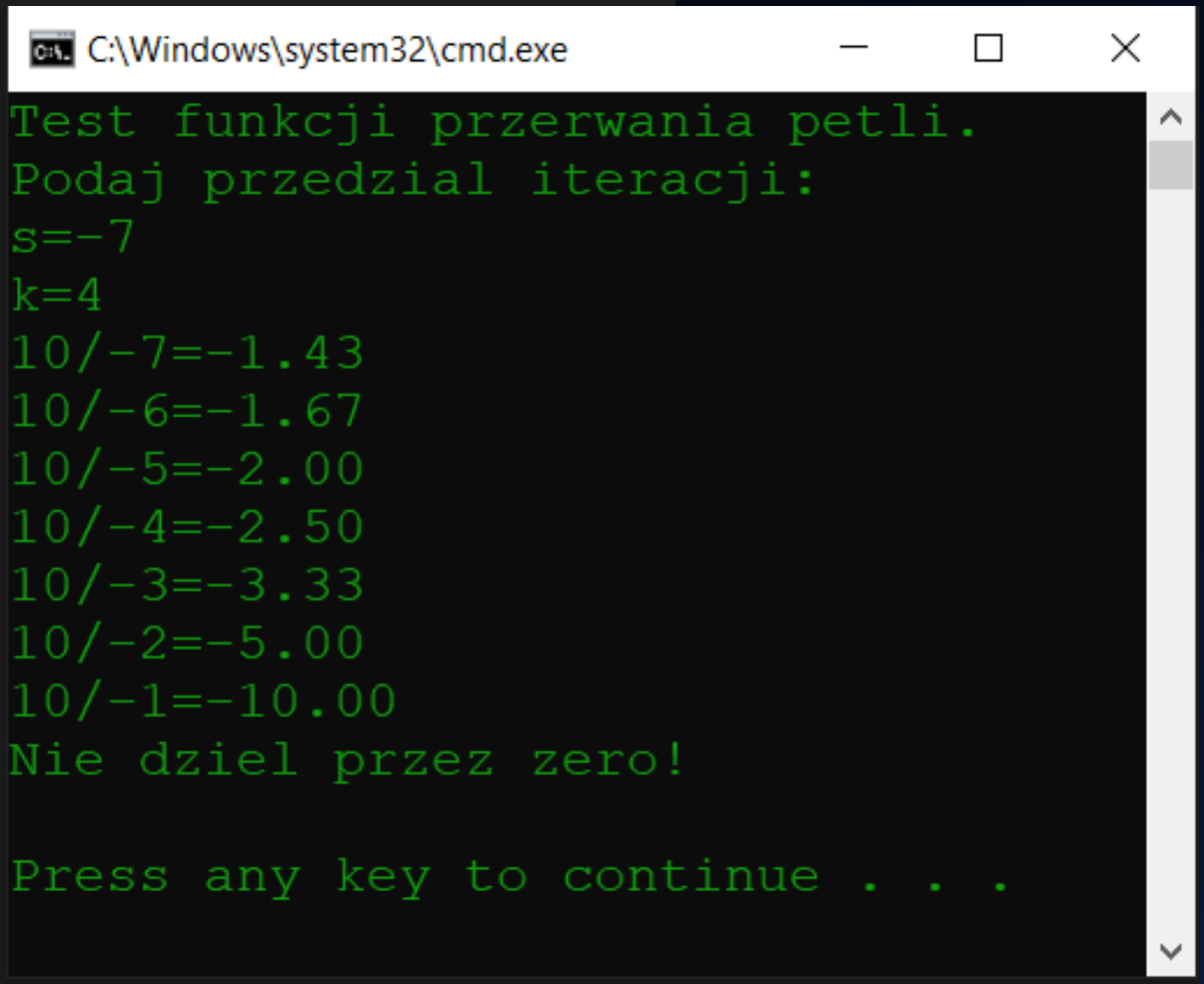
```
1   #include
2   #include
3
4   int main
5   {
6       prin
7       prin
8       prin
9       int
10      scan
11      prin
12      scan
13      for(
14      {
15
16
17      }
```

```
C:\Windows\system32\cmd.exe
Test funkcji przerwania petli.
Podaj przedzial iteracji:
s=6
k=-4
10/6=1.67
10/5=2.00
10/4=2.50
10/3=3.33
10/2=5.00
10/1=10.00
10/-1=-10.00
10/-2=-5.00
10/-3=-3.33
10/-4=-2.50

Press any key to continue . . .
```

```
10/8=1.25

Press any key to continue . . .
```

Przerwanie pętli

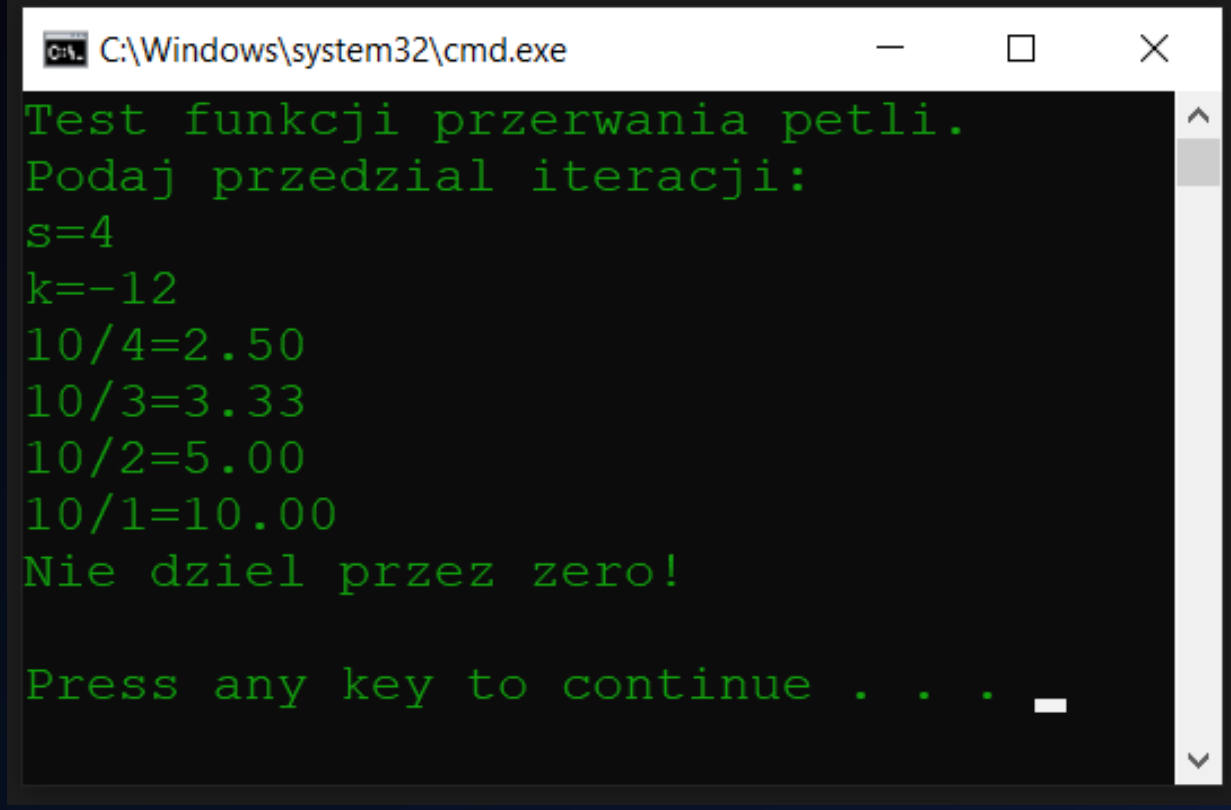
```
1  #include <stdio.h>
2  #include <math.h>
3
4  int main()
5  {
6      printf("Test fun
7      printf("Podaj pr
8      printf("s=");
9      int s,k;
10     scanf("%i",&s);
11     printf("k=");
12     scanf("%i",&k);
13     for(int i=s;s<k?
14     {
15         if(i==0){
16             printf("
17             break;
18         }
19         printf("10/%
20     }
21 }
```



Przerwanie pętli

- Prz… …kże za pon…
- Ozn… …czenie biez…

```c
#include <stdio.h>
#include <math.h>

int main()
{
    printf("Test funkcji przerwania petli.\n");
    printf("Podaj przedzial iteracji:\n");
    printf("s=");
```



Pętle warunkowe

- Pętle warunkowe powtarzają przypisany do niej kod gdy warunek w niej zdefiniowany zwraca wartość TRUE
- Konstrukcja pętli warunkowych jest modyfikacja instrukcji warunkowej `if`
- W języku C dostępne są dwie konstrukcje warunkowe
- Analizująca warunek przed wykonaniem kodu - `while`
- Analizująca warunek po wykonaniu kodu – `do..while`

Pętla warunkowe - while

```
while(warunek)
{
    kod;
}
```

- Zasady budowania warunku są analogiczne do tych znanych z instrukcji warunkowej `if`
- Zazwyczaj zmienne analizowane w warunku są przetwarzane w kodzie pętli
- Dla pętli `while` możliwa jest sytuacja że kod niezostanie nawet raz wykonany

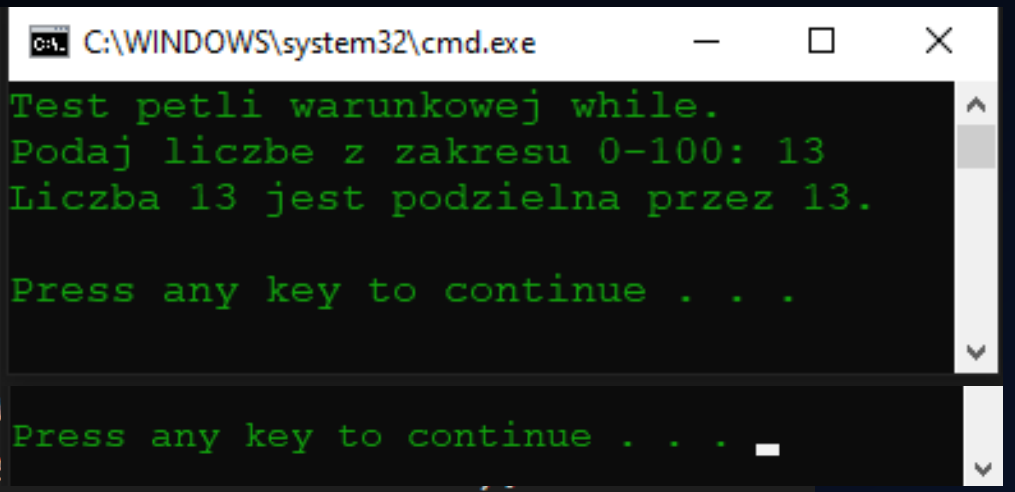
Pętla warunkowa – do..while

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

- W układzie pętli warunkowej `do..while` kod pętli zostanie wykonany przynajmniej raz
- Nie można zbudować warunku w oparciu o zmienną zadeklarowaną w bloku petli

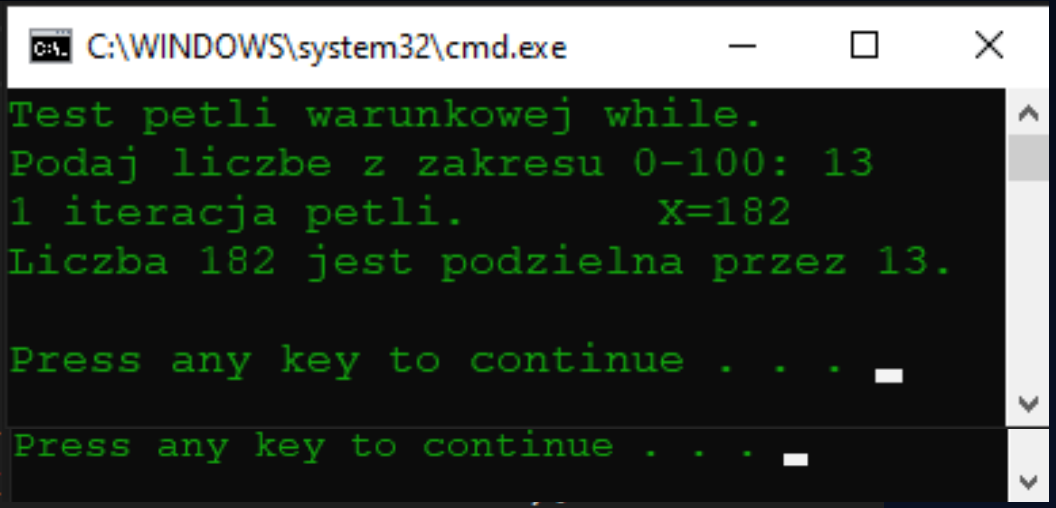
Przykład

```c
#include <stdio.h>
#include <time.h>
#include <stdlib.h>

int main()
{
    printf("Test petli w
    printf("Podaj liczbe
    unsigned long long x;
    scanf("%llu",&x);
    while(x%13!=0)
    {
        static int i=0;
        x=x*(x+1);
        i++;
        printf("%i iteracja petli.\tX=%llu\n",i,x);
    }
    printf("Liczba %llu jest podzielna przez 13.\n",x);
}
```



Przykład

```c
#include <stdio.h>
#include <time.h>
#include <stdlib.h>

int main()
{
    printf("Test petli
    printf("Podaj licz
    unsigned long long x;
    scanf("%llu",&x);
    do
    {
        static int i=0;
        x=x*(x+1);
        i++;
        printf("%i iteracja petli.\tX=%llu\n",i,x);
    }while(x%13!=0);
    printf("Liczba %llu jest podzielna przez 13.\n",x);
}
```



Przykład

```c
#include <stdio.h>
#include <time.h>
#include <stdlib.h>

int main()
{
    printf("Test petli warunkowej while.\n");
    printf("Podaj liczbe z zakresu 0-100: ");
    unsigned long long x;
    scanf("%llu",&x);
    do
    {
        static int i=0;
        x=x*(x+1);
        i++;
        printf("%i iteracja petli.\tX=%llu\n",i,x);
    }while(i<5);
    printf("Liczba %llu jest podzielna przez 13.\n",x);
}
```

Zadanie

```
1  #include <stdio.h>
2  float modul(float);
3  int main()
4  {
```

C:\WINDOWS\system32\cmd.exe

```
Program oblicza pierwiastek kwadratowy podanej liczby rzeczywistej.
Okresl dokladnosc obliczen: 0.000001
Podaj liczbe: 369.789
185.395
93.6946
48.8207
28.1975
20.6559
19.2791
19.23
19.2299
Wartosc obliczono po 8 krokach.
Pierwiastek kwadratowy z 369.789 wynosi: 19.2299

Press any key to continue . . .
```

```
22 {
23     if(x<0)
24         x=-x;
25     return x;
26 }
```