

## Podprogramy

Wstęp do Informatyki i Programowania

Maciek Gębala

16 października 2025

Maciek Gębala Podprogramy

## Podprogramy

Jednym z pierwszych kroków aby pisać sprawnie programy jest ich strukturyzacja, czyli podział na podprogramy (funkcje i procedury).

Procedura to wyodrębniona część programu, wykonująca się na podstawie podanych parametrów i nie zwracająca wartości (nie może być traktowana jako wyrażenie).

Funkcja to wyodrębniona część programu, wykonująca się na podstawie podanych parametrów i zwracająca jakąś wartość (może być traktowana jako wyrażenie).

Zakładamy, że dobrze napisany podprogram nie zależy od wartości/zmiennych zewnętrznych (globalnych), czyli nie będących parametrami wywołania.

Maciek Gębala Podprogramy

## Język C

Formalnie w języku C definiujemy tylko funkcje, procedury to funkcje zwracające typ `void` (pusty). Wartości funkcji nie muszą być odbierane.

### Definiowanie funkcji

```
<typ wyniku> <nazwa>(<lista parametrów>) { <implementacja>
};
```

### Zwracanie wyniku

Wynik zwracamy korzystając z kluczowego słowa `return` (które jednocześnie przerywa wykonanie funkcji).

W C nie można definiować funkcji w funkcjach.

Funkcja `int main(...)` jest zawsze w C wywoływana domyślnie jako początek programu.

Maciek Gębala Podprogramy

## Język C

### Przekazywanie parametrów

W C parametry przekazuje się przez kopię wartości - zmiany wewnątrz funkcji nie są przekazywane na zewnątrz.

Jeśli chcemy zmieniać wartość parametrów to przekazujemy w parametrach adres zmiennej i wykonujemy operacje na wskazaniach tego adresu (operatory: `&` do pobrania adresu i `*` do wskazania adresu, zobacz `scanf`)

Więcej szczegółów będzie na wykładach o złożonych i dynamicznych typach danych.

Maciek Gębala Podprogramy

Notatki

Notatki

Notatki

Notatki

## Przykład w języku C

```
gcdtest.c
1 #include <stdio.h>
2 void read(int *a, int *b) {
3     printf("Podaj pierwszą liczbę: ");
4     scanf("%d", a);
5     printf("Podaj drugą liczbę: ");
6     scanf("%d", b);
7 }
8 int gcd(int a, int b) {
9     int c;
10    while ( b != 0 ) {
11        if ( a < b ) {
12            c = a;
13            a = b;
14            b = c;
15        } else {
16            a = a - b;
17        }
18    }
19    return a;
20 }
```

Maciek Gębala Podprogramy

Notatki

## Przykład w języku C

```
gcdtest.c
21 void write(int a) {
22     printf("Największy wspólny dzielnik to %d\n", a);
23 }
24 int main() {
25     int a, b, c;
26     read(&a, &b);
27     c = gcd(a, b);
28     write(c);
29
30     return 0;
31 }
```

Maciek Gębala Podprogramy

Notatki

## Język Ada

W Adzie mamy osobne słowa kluczowe do definiowania procedur i funkcji.

### Definiowanie procedury

procedure <nazwa>(lista parametrów) is <implementacja>

### Definiowanie funkcji

function <nazwa>(lista parametrów) return <typ> is  
<implementacja>

### Zwracanie wyniku

Wynik zwracamy korzystając z kluczowego słowa return (które jednocześnie przerywa wykonanie funkcji).

W można definiować procedury/funkcje wewnątrz procedur i funkcji.

Należy pamiętać, że program w Adzie jest zawsze jedną procedurą.

Maciek Gębala Podprogramy

Notatki

## Język Ada

### Przekazywanie parametrów

W Adzie parametry przekazuje się przez referencję na trzy sposoby:

- 1 IN – tryb domyślny, wewnątrz podprogramu parametry traktowane jako stałe;
- 2 OUT – tryb z możliwością przekazania wartości na zewnątrz ale traktowany jak niezainicjowany wewnątrz;
- 3 IN-OUT – pełny tryb referencji, inna nazwa zmiennej podanej w wywołaniu.

Maciek Gębala Podprogramy

Notatki

## Przykład w języku Ada

```
gcdtest.adb
1  with Ada.Text_IO; use Ada.Text_IO;
2  with Ada.Integer_Text_IO; use Ada.Integer_Text_IO;
3  procedure gcdtest is
4    procedure read(a : out Integer; b : out Integer) is
5      begin
6        Put("Podaj pierwszą liczbę: ");
7        Get(a);
8        Put("Podaj drugą liczbę: ");
9        Get(b);
10   end read;
```

Maciek Gębala Podprogramy

Notatki

## Przykład w języku Ada

```
gcdtest.adb
11  function gcd(x : in Integer; y : in Integer) return Integer is
12    a, b, c : Integer;
13  begin
14    a := x;
15    b := y;
16    while b /= 0 loop
17      if a < b then
18        c := a;
19        a := b;
20        b := c;
21      else
22        a := a - b;
23      end if;
24    end loop;
25    return a;
26  end gcd;
```

Maciek Gębala Podprogramy

Notatki

## Przykład w języku Ada

```
gcdtest.adb
27  procedure write(c : in Integer) is
28  begin
29    Put_Line("Największy wspólny dzielnik to " & c'Image);
30  end write;
31  a, b, c : Integer;
32  begin
33    read(a, b);
34    c := gcd(a, b);
35    write(c);
36  end gcdtest;
```

Maciek Gębala Podprogramy

Notatki

## Język Python

W Pythonie podprogramy definiujemy słowem kluczowym `def`

Definiowanie podprogramu

```
def <nazwa>(lista parametrów) :    <implementacja>
```

Zwracanie wyniku

Wynik zwracamy korzystając z kluczowego słowa `return` (które jednocześnie przerywa wykonanie funkcji). Podprogram nie ma w deklaracji, że jest funkcją.

W Pythonie można definiować podprogramy wewnątrz podprogramów.

Maciek Gębala Podprogramy

Notatki

### Przekazywanie parametrów

Proste typy są przekazywane przez wartość a złożone przez referencję (i te są modyfikowalne na zewnątrz).

### Przykład w języku Python

gcdtest.py

```
1 def read() :
2     a = int(input("Podaj pierwszą liczbę: "))
3     b = int(input("Podaj drugą liczbę: "))
4     return a, b
5
6 def gcd( a, b ) :
7     while b != 0 :
8         if a < b :
9             a, b = b, a
10        else :
11            a = a - b
12        return a
13
14 def write( c ) :
15     print(f"Największy wspólny dzielnik to {c}")
16
17 def main():
18     a,b = read()
19     c = gcd( a, b )
20     write( c )
21
22 if __name__ == "__main__":
23     main()
```

### Przykład w języku Python (2)

gcdtest2.py

```
1 def main():
2     def read() :
3         a = int(input("Podaj pierwszą liczbę: "))
4         b = int(input("Podaj drugą liczbę: "))
5         return a, b
6
7     def gcd( a, b ) :
8         while b != 0 :
9             if a < b :
10                a, b = b, a
11            else :
12                a = a - b
13            return a
14
15     def write( c ) :
16         print(f"Największy wspólny dzielnik to {c}")
17
18     a,b = read()
19     c = gcd( a, b )
20     write( c )
21
22 if __name__ == "__main__":
23     main()
```

### Rekurencja

Wszystkie trzy języki pozwalają na definiowanie wywołań rekurencyjnych, czyli wywołanie podprogramu wewnątrz własnej definicji.

Należy zadbać, aby nie spowodować nieskończonego ciągu odwołań rekurencyjnych.

W praktyce unika się wywołań rekurencyjnych, zastępując je iteracjami w pętli.

## Notatki

## Notatki

## Notatki

## Przykład rekurencji - liczby Fibonacciego

### Rekurencyjna definicja

$Fibonacci(0) = 0$   
 $Fibonacci(1) = 1$   
 $Fibonacci(n) = Fibonacci(n-1) + Fibonacci(n-2)$  dla  $n > 1$

Maciek Gębala Podprogramy

## Implementacja w języku C

```
fib.c
1 #include <stdio.h>
2 unsigned fibonacci(unsigned n) {
3     if ( n <= 1 ) return n;
4     return fibonacci(n-1) + fibonacci(n-2);
5 }
6 int main() {
7     unsigned n, m;
8
9     printf("Podaj liczbę:");
10    scanf("%u", &n);
11    m = fibonacci(n);
12    printf("Fib(%u)=%u\n", n, m);
13
14    return 0;
15 }
```

Maciek Gębala Podprogramy

## Implementacja w języku Ada

```
fib.adb
1 with Ada.Text_IO; use Ada.Text_IO;
2 with Ada.Integer_Text_IO; use Ada.Integer_Text_IO;
3 procedure fib is
4     function fibonacci(n : in Natural) return Natural is
5     begin
6         if n<=1 then
7             return n;
8         end if;
9         return fibonacci(n-1) + fibonacci(n-2);
10    end fibonacci;
11    n, m : Natural;
12 begin
13     Put("Podaj liczbę:");
14     Get(n);
15     m := fibonacci(n);
16     Put_Line("Fib(" & n'Image & ")=" & m'Image);
17 end fib;
```

Maciek Gębala Podprogramy

## Implementacja w języku Python

```
fib.py
1 def fibonacci( n ) :
2     if n <= 1 :
3         return n
4     return fibonacci(n-1) + fibonacci(n-2)
5
6 def main():
7     n = int(input("Podaj liczbę:"))
8     m = fibonacci(n)
9     print(f"Fib({n})={m}")
10
11 if __name__ == "__main__":
12     main()
```

Maciek Gębala Podprogramy

Notatki

Notatki

Notatki

Notatki