PODSTAWY PROGRAMOWANIA

dr hab. Mariusz Meszka

Akademia Górniczo-Hutnicza w Krakowie

http://home.agh.edu.pl/∼meszka

Program wykładu

1. Podstawowe pojęcia i definicje. Metody zapisu algorytmów.
2. Język programowania C: wprowadzenie, typy i rozmiary danych, operatory i wyrażenia.
3. Język programowania C: składnia języka – instrukcje, struktura programu.
4. Język programowania C: funkcje, preprocesor, klasy pamięci.
5. Język programowania C: operacje na wskaźnikach i adresach, tablice dynamiczne.
6. Język programowania C: wejście i wyjście, pliki.
7. Język programowania C: podstawowe biblioteki i narzędzia, przetwarzanie kodu źródłowego w różnych systemach operacyjnych.

Program wykładu (cd.)

8. Maszynowa reprezentacja informacji (systemy zapisu liczb i arytmetyka).
9. Elementy teorii złożoności algorytmów: notacja, złożoność pesymistyczna i średnia, klasyfikacja problemów.
10. Rekurencja i iteracja – porównanie metod.
11. Wybrane algorytmy sortowania: proste wstawianie, proste wybieranie, bąbelkowe, szybkie, scalanie, stogowe.
12. Sortowanie w czasie linowym: przez zliczanie i kubełkowe. Wyszukiwanie mediany.
13. Złożone struktury danych: stosy, kolejki, listy pojedynczo wiązane.
14. Listy podwójnie i wielokrotnie wiązane.
15. Drzewa – implementacje, porządki w drzewach. Drzewa przeszukiwań binarnych.

Literatura

[1] B. Kernighan, D.M. Ritchie, Język ANSI C. Programowanie. Wydanie II , Helion 2020.
[2] T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, Wprowadzenie do algorytmów, PWN 2012.
[3] N. Wirth, Algorytmy + struktury danych = programy, WNT 2002.

Podstawowe obiekty danych

(1) stałe
(2) zmienne

Rodzaje stałych

(1) liczbowe
(2) znakowe
(3) tekstowe

Typy danych

(1) proste
  (a) liczbowe
    (i) całkowite: **int**
    (ii) zmiennopozycyjne: **float, double**
  (b) znakowe: **char**
  (c) logiczne
  (d) wyliczeniowe: **enum**
(2) złożone
  (a) tablice
  (b) struktury
  (c) unie

Modyfikatory

short, **long**, **signed**, **unsigned**

Definicja stałej

const *typ nazwastałej* = *wartość*;

Deklaracja zmiennej

typ nazwazmiennej;
typ nazwazmiennej = *wartość*;

Operatory

(1) arytmetyczne:

$+ \quad - \quad * \quad / \quad \% \quad ++ \quad --$

(2) relacyjne:

$< \quad > \quad <= \quad >= \quad == \quad !=$

(3) logiczne

&& || !

(4) przypisania

$= \quad$ *operator*$=$

(5) bitowe

& | ^ << >> ~

(6) rozmiaru

sizeof(*zmienna***)**

sizeof(*typ***)**

(7) warunkowy

wyrażenie1 **?** *wyrażenie2* **:** *wyrażenie3*

Instrukcje warunkowe

(1)

if (*warunek*)
  instrukcja;

(2)

if (*warunek*)
  instrukcja1;
else
  instrukcja2;

Instrukcje warunkowe cd.

(3)

```
switch (wyrażenie)
  {
   case przypadek1: instrukcja1;
                    break;
   case przypadek2: instrukcja2;
                    break;
    ⋮
   case przypadekN: instrukcjaN;
                    break;
   default: instrukcja;
  }
```

Instrukcje iteracyjne

(1)

while (*warunek*)
  instrukcja;

(2)

do
  instrukcja;
while (*warunek*);

(3)

for (*wyrażenie1*; *wyrażenie2*; *wyrażenie3*)
  instrukcja;

Instrukcje skoku

(1)

break;

(2)

continue;

(3)

return *wyrażenie* **;**

(4)

goto *etykieta* **;**

Blok instrukcji

{
instrukcja1;
instrukcja2;
⋮
instrukcjaN;
}

Instrukcja pusta

;

Funkcja to zamknięty w całość fragment kodu programu, który jest wykonywany po jego jawnym wywołaniu. Do funkcji można przekazywać parametry, funkcja może zwracać wartość.

Definicja funkcji

typzwracanejwartości nazwafunkcji(*deklaracjeparametrów*)
{
ciąg deklaracji i instrukcji
return *wyrażenie*;
}

Struktura programu

dyrektywy preprocesora
definicje i deklaracje obiektów zewnętrznych
definicje funkcji

Dyrektywy preprocesora

#**include** <*nazwapliku*>
#**include** "*nazwapliku*"
#**define** *identyfikator ciągznaków*
#**undef** *identyfikator*
#**if** *wyrażenie1 ...* #**endif**

Standardowe wejście i wyjście

int getchar(void)
int putchar(int)
int printf(char *format, *argument1, argument2, ...***)**
int scanf(char *format, *wskaźnik1, wskaźnik2, ...***)**

format:

%<*flaga*><*szerokość*><.*precyzja*><*modyfikator*>*typ*

typ:

d,i	**int**
u	**unsigned int**
o	**int** w formacie ósemkowym
x	**int** w formacie szesnastkowym
o	**int** w formacie ósemkowym
c	**char**
s	**char***
f,e	**double**