Podstawy programowania komputerów

Wykład 3: „Dane w ANSI C”

swalover@ie.tu.koszalin.pl

Dane: stałe i zmienne

- Stałe (constans) są ustalone przed uruchomieniem programu i pozostają niezmienne cały czas jego działania.
- Zmienne (variables) są to dane, które mogą się zmieniać w trakcie działania programu.

Różnica polega na tym, że zmiennej można przypisywać rozmaite wartości w trakcie działania programu, natomiast w przypadku stałej nie jest to możliwe.

Słowa kluczowe typów danych

Pierwotne w K&R C

- int
- long
- short
- unsigned
- char
- float
- double

Dodatkowe w ANSI C

- signed
- void
- const
- volatile

Typy całkowite a zmiennoprzecinkowe

- W C/C++ liczby **całkowite** nigdy nie zawierają kropki dziesiętnej i są przechowywane w pamięci w postaci binarnej: 7 ⇨ 00000111
- Liczby **zmiennoprzecinkowe** (mat. **rzeczywiste**) w pamięci rozbite są na dwie części – ułamkową i wykładniczą: 7,0 ⇨ 0.7 i 1

Wartości zmiennoprzecinkowe są często jedynie przybliżeniami prawdziwych wartości rzeczywistych

Stałe liczbowe

zapis matematyczny		zapis w C/C++
15,183	⇨	15.183
$3,4*10^{-38}$	⇨	3.4E-38
10^{12}	⇨	1E12
-10^{-6} lub -0,000001	⇨	-1E-6

Zmienne

- Każda zmienna użyta w programie musi być zadeklarowana. Deklaracja określa typ danych jakie zmienna może przechowywać.
- Deklaracja zmiennych

```
int nr_1, nr_2, nr_3;
char Imie[];
float WagaMoja;
```

Wejście – wyjście

- Program C/C++ komunikuje się z otoczeniem za pomocą strumieni. Strumień zawiera dane w postaci ciągu bajtów, a funkcje operujące na tych bajtach muszą "umieć" je zinterpretować.

- Funkcje te i strumienie nie zostały zdefiniowane w rdzeniu języka C, lecz w bibliotekach. Prototypy funkcji (nagłówki) są udostępnione przez pliki nagłówkowe

Wejście – wyjście

- Strumienie standardowe:

stdin - standardowy strumień wejściowy połączony z klawiaturą;

stdout - standardowy strumień wyjściowy połączony z monitorem.

- Strumienie standardowe dostępne są poprzez plik nagłówkowy <stdio.h>, plik ten dołącza się odpowiednią dyrektywą preprocesora:

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

Wejście – wyjście

Funkcje obsługi wejścia-wyjścia:

- getchar( ) - pobierz jeden znak z stdin
- putchar (<znak>) - wyprowadź znak do stdout
- gets( ) / puts( ) - wprowadzenie / wyprowadzenie łańcucha znaków
- scanf( ) / printf( ) - wprowadzenie / wyprowadzenie danych z formatowaniem

Funkcje te są również dostępne poprzez plik nagłówkowy <stdio.h>

Wejście – wyjście w C++

- plik nagłówkowy <iostream.h> udostępnia strumienie

cin ⇨ strumień wejściowy

cout ⇨ strumień wyjściowy

- dla strumieni zdefiniowane są operatory wprowadzania i wyprowadzania :

>> ⇨ wczytaj ze strumienia cin

<< ⇨ wyprowadź do strumienia cout

Przykład: `cout << "Dzień dobry\n";`

Wyświetlanie danych na ekranie

Funkcja printf ( ) jest jedną z najczęściej używanych funkcji języka podstawowego, jej wywołanie ma ogólną formę:

```
printf(<format>,<arg1>,<arg2>,...);
```

<format> jest wzorcem wyprowadzania argumentów arg1, arg2, ...

Specyfikatory formatu

Format wyprowadzania danych składa się ze specyfikatorów (wzorców konwersji), które umożliwiają kontrolę nad wyprowadzanym komunikatem

Specyfikatory formatu(c.d.)

Specyfikator	⇨	Opis
%s	⇨	łańcuch znaków - tekst (string)
%c	⇨	pojedynczy znak (character)
%d	⇨	liczba całkowita (typu int) w postaci dziesiątkowej (decimal)
%o	⇨	liczba całkowita (typu int) w postaci ósemkowej (octal)
%x	⇨	liczba całkowita (typu int) w postaci szesnastkowej (hexadecimal)
%f	⇨	liczba rzeczywista (typu float) w postaci dziesiątkowej (floating point)

Specyfikatory formatu(c.d.)

Specyfikator	⇨	Opis (c.d.)
%u	⇨	liczba bez znaku (typu unsigned)
%ld	⇨	liczba całkowita długa (typu long int) w postaci dziesiątkowej (decimal)
%Lf	⇨	liczba rzeczywista poczwórnej precyzji (typu long double float) w postaci dziesiątkowej (floating point)
%e	⇨	liczba w formacie wykładniczym
%g	⇨	automatyczny wybór formatu %f lub %e (decyduje zwięzłość zapisu)

Sekwencje sterujące

W cudzysłowach oprócz tekstu i wzorców konwersji spotkać można także tzw. sekwencje ucieczki (escape seqences). Są to symbole zaczynające się od znaku \ (ukośnik, backslash).

Sekwencja	Znaczenie	Wartość dziesiątkowa
\a	alarm (bell, BEL)	7
\b	cofnięcie karetki (backspace, BS)	8
\e	znak ucieczki (escape, ESC)	27
\f	nowa strona (form feed, FF)	12

Sekwencje sterujące (c.d.)

Sekwencja	Znaczenie	Wartość dziesiątkowa
\n	nowa linia (new line, NL)	10
\r	powrót karetki (carriage return, CR)	13
\t	tabulacja pozioma (horizontal tab, HT)	9
\v	tabulacja pionowa (vertical tab, VT)	11
\\	ukośnik (backslash)	
\"	cudzysłów (double quote)	
\'	apostrof (single quote)	
\<ENTER>	kontynuacja linii (line continuation)	
\nnn nnn =	wartość ósemkowa znaku	

Operatory arytmetyczne

+	operator dodawania
-	operator odejmowania
*	operator mnożenia
/	operator dzielenia
%	operator dzielenia modulo - wyznacza resztę z dzielenia, może być stosowany tylko do argumentów typu całkowitego

np.: `(5%2)` zwraca wartość `1`

Operatory arytmetyczne (c.d.)

`++`	operator zwiększania (inkrementacji)
`--`	operator zmniejszania (dekrementacji)

Uważać, bo mogą być stosowane przedrostkowo lub przyrostkowo!

PRZYKŁAD: n=5;

`k=n++;`	k=n; n=n+1; //k=5, n=6
`k=++n;`	n=n+1; k=n; //n=6, k=6

Operatory przypisania

Zapis:

```
<zmienna> <operator>=<wyrażenie>
```

jest równoważny zapisowi

```
<zmienna>=<zmienna> <operator>(<wyrażenie>)
```

Przykłady:

Styl C

```
index+=2;
x/=y;
x*=y+1;
```

Styl Pascal'a

```
index=index+2;
x=x/y;
x=x*(y+1);
```

Wyrażenia arytmetyczne

Przy obliczaniu wartości wyrażenia arytmetycznego może mieć znaczenie kolejność obliczeń przyjęta w C/C++:

( ) ⇨ ++ -- ⇨ * / % ⇨ + - ⇨ = += -= *= /= %=

Przykłady:

Matematyka	C/C++
$\frac{2x-1}{ax^2}$	`(2*x-1)/a/x/x;`
$3(x^2+ax-y)$	`3*(x*x+a*x-y);`