Procedūrinio programavimo pagrindai

Baziniai elementai

lekt. Irmantas Radavičius
irmantas.radavicius@mif.vu.lt
Informatikos institutas, MIF, VU

Turinys

Įvadas

Bitų seka

- 0100 0001

Reikšmės

- 1 baitas: $2^8 = 256$ skirtingų reikšmių
- 2 baitai: $2^{16} = 65536$ skirtingų reikšmių
- …

Interpretacija

- ‘A’
- 65
- 4.0625
- …

Duomenų tipo apibrėžimas

Duomenų tipo sąvoka

- duomenų prasmė
- reikšmių saugojimo būdas ir diapazonas
- taikytinos operacijos

C duomenų tipai

Primityvūs duomenų tipai

- simbolis (char)
- sveikasis skaičius (int)
- realus skaičius (float, double)

Išvestiniai duomenų tipai

- rodyklė
- funkcija
- masyvas
- struktūra
- ...

C duomenų tipai

Sveikųjų skaičių dydžio modifikatoriai

- short int, short (bent 16 bitų)
- long int, long (bent 32 bitų)
- sizeof(short) <= sizeof(int) <= sizeof(long)

Sveikųjų skaičių ženklo modifikatoriai

- signed int, signed, unsigned int, unsigned
- signed long int, unsigned short, etc.
- signed char, unsigned char

C duomenų tipai

Tipinė sveikųjų tipų implementacija

Type	Bytes	Bits	Range
short int	2	16	-32,768 -> +32,767
unsigned short int	2	16	0 -> +65,535
unsigned int	4	32	0 -> +4,294,967,295
int	4	32	-2,147,483,648 -> +2,147,483,647
long int	4	32	-2,147,483,648 -> +2,147,483,647
signed char	1	8	-128 -> +127
unsigned char	1	8	0 -> +255

C duomenų tipai

Slankaus kablelio formatai

- float
- double
- long double
- sizeof(float) <= sizeof(double) <= sizeof(long double)

Type specifiers	Precision (decimal digits)		Exponent range	
	Minimum	IEEE 754	Minimum	IEEE 754
`float`	6	7.2 (24 bits)	±37	±38 (8 bits)
`double`	10	15.9 (53 bits)	±37	±307 (11 bits)
`long double`	10	34.0 (113 bits)	±37	±4931 (15 bits)

Kintamieji

- kintamojo aprašas
- kintamojo inicializacija
- kintamojo panaudojimas

main.c

```c
/* Variable declarations */

int main(){

   int n;
   int x = 5, y;
   double d1, d2 = x;
   unsigned u = 1;

   return 0;
}
```

Kintamųjų vardai

Taisyklės

- vardas prasideda raide arba apatiniu brūkšneliu

Rekomendacijos

- pradėti mažąja raide
- parinkti prasmingą vardą
- netrumpinti
- rašyti angliškai
- atskirti žodžius

auto	double	int	struct
break	else	long	switch
case	enum	register	typedef
char	extern	return	union
const	float	short	unsigned
continue	for	signed	void
default	goto	sizeof	volatile
do	if	static	while

Reikšmės

Int

- 1234

Long

- 1234L arba 1234l
- 1234UL arba 1234ul
- Konstanta netelpa į int rėžius

Skaičiavimo sistemos

- 10-ainė, pvz. 1234
- 8-ainė, pvz. 01234
- 16-ainė, pvz. 0x1234

Reikšmės

Char

- ‘0’
- ‘\060’
- ‘\x30’

`\a`	alert (bell) character	`\\`	backslash
`\b`	backspace	`\?`	question mark
`\f`	formfeed	`\'`	single quote
`\n`	newline	`\"`	double quote
`\r`	carriage return	`\`*ooo*	octal number
`\t`	horizontal tab	`\x`*hh*	hexadecimal number
`\v`	vertical tab		

Reikšmės

Double

- 1.04
- 1.1e1

Float

- 1.04F arba 1.04f

Operacija

Pagrindinės sąvokos

- operacija
- operatorius
- operandas

Operatorių tipai

- dviviečiai (angl. binary)

  a + b
- vienviečiai (angl. unary)

  a + (-b)

Operatoriai

Ženklo operatoriai (2)

- vienviečiai + -

Aritmetiniai operatoriai (5)

- dviviečiai + -
- * / %

Priskyrimo operatorius (6)

- =
- += -= *= /= %=

Didinimo ir mažinimo operatoriai

- ++x --x
- x++ x--

Operatoriai

Palyginimo operatoriai (6)

- < > <= >=
- ar lygu ==
- ar nelygu !=

Loginiai operatoriai (3)

- neigimas !
- konjunkcija &&
- disjunkcija II

Operatoriai

Bitiniai operatoriai (6)

- not ~
- and &
- or |
- xor ^
- left shift <<
- right shift >>

Priskyrimo operatoriai (5)

- &= |= ^= <<= >>=

Operatorių asociatyvumas

Operatorių asociatyvumas

Kairysis asociatyvumas

- (a operatorius b) operatorius c

  a + b + c

Dešinysis asociatyvumas

- a operatorius (b operatorius c)

  a = b = c

Operatorių prioritetai

OPERATORS	ASSOCIATIVITY
`()  []  ->  .`	left to right
`! ~ ++ -- + - * & (type) sizeof`	right to left
`*  /  %`	left to right
`+  -`	left to right
`<<  >>`	left to right
`<  <=  >  >=`	left to right
`==  !=`	left to right
`&`	left to right
`^`	left to right
`|`	left to right
`&&`	left to right
`||`	left to right
`?:`	right to left
`= += -= *= /= %= &= ^= |= <<= >>=`	right to left
`,`	left to right

Išvedimas ir įvedimas

Standartinė biblioteka <stdio.h>

- printf

```
int printf(char *format, arg1, arg2, ...)
```

- scanf

```
int scanf(char *format, ...)
```

Printf formato specifikacija

CHARACTER	ARGUMENT TYPE; PRINTED AS
d, i	int; decimal number.
o	int; unsigned octal number (without a leading zero).
x, X	int; unsigned hexadecimal number (without a leading 0x or 0X), using abcdef or ABCDEF for 10, ..., 15.
u	int; unsigned decimal number.
c	int; single character.
s	char *; print characters from the string until a '\0' or the number of characters given by the precision.
f	double; [−]*m.dddddd*, where the number of *d*'s is given by the precision (default 6).
e, E	double; [−]*m.dddddd*e±*xx* or [−]*m.dddddd*E±*xx*, where the number of *d*'s is given by the precision (default 6).
g, G	double; use %e or %E if the exponent is less than −4 or greater than or equal to the precision; otherwise use %f. Trailing zeros and a trailing decimal point are not printed.
p	void *; pointer (implementation-dependent representation).
%	no argument is converted; print a %.

Scanf formato specifikacija

CHARACTER	INPUT DATA; ARGUMENT TYPE
`d`	decimal integer; `int *`.
`i`	integer; `int *`. The integer may be in octal (leading `0`) or hexadecimal (leading `0x` or `0X`).
`o`	octal integer (with or without leading zero); `int *`.
`u`	unsigned decimal integer; `unsigned int *`.
`x`	hexadecimal integer (with or without leading `0x` or `0X`); `int *`.
`c`	characters; `char *`. The next input characters (default 1) are placed at the indicated spot. The normal skip over white space is suppressed; to read the next non-white space character, use `%1s`.
`s`	character string (not quoted); `char *`, pointing to an array of characters large enough for the string and a terminating `'\0'` that will be added.
`e, f, g`	floating-point number with optional sign, optional decimal point and optional exponent; `float *`.
`%`	literal `%`; no assignment is made.

I’ll be back...