Uvod u programiranje

znakovni i logički tip podataka

Prikaz slova i ostalih znakova

- Kombinacijom jedinica i nula – kôdom
- Koliko ima znakova?
  - 26 velikih slova engleske abecede A - Z
  - 26 malih slova engleske abecede a - z
  - 10 znamenaka 0 - 9
  - operatori, interpunkcije, upravljački znakovi
- Za prikaz je dovoljan 1 oktet
- ASCII (ISO-7 standard): 7 bita za informaciju + 1 bit za *paritet*
  ⇨ $2^7 = 128$ različitih znakova
  - ASCII - American Standard Code for Information Interchange
  - ISO - International Organization for Standardization.
- *Paritet*: ako je u informaciji neparan broj bita, bit pariteta postavlja se na 1, inače na 0 (može i obratno: *odd/even parity*). Omogućuje otkrivanje jednostruke pogreške pri prijenosu informacija

Tablica ASCII kontrolnih znakova koji se ne mogu ispisati

Dec. broj	C konst.	Znak	Dec. broj	Znak
0	'\0'	Nul znak (NULL)	16	znak prekida veze (DLE)
1		početak zaglavlja (SOH)	17	provjera uređaja 1 (DC1)
2		početak teksta (STX)	18	provjera uređaja 2 (DC2)
3		kraj teksta (ETX)	19	provjera uređaja 3 (DC3)
4		kraj prijenosa (EOT)	20	provjera uređaja 4 (DC4)
5		kraj upita (ENQ)	21	negativna potvrda (NAK)
6		Potvrda (ACK)	22	sinkrono mirovanje (SYN)
7	'\a'	Alarm (BEL)	23	kraj prijenosnog bloka (ETB)
8	'\b'	Backspace (BS)	24	otkaži (CAN)
9	'\t'	vodoravni tabulator (HT)	25	kraj medija (EM)
10	'\n'	sljedeći red/novi red (LF)	26	Zamjena (SUB)
11	'\v'	okomiti tabulator (VT)	27	Escape (ESC)
12	'\f'	nova stranica (FF)	28	razdjelnik datoteka (FS)
13	'\r'	skok na početak reda (CR)	29	razdjelnik grupe (GS)
14		pomak van (SO)	30	razdjelnik zapisa (RS)
15		pomak unutra (SI)	31	razdjelnik jedinice (US)

Tablica ASCII znakova koji se mogu ispisati

Dec. broj	Znak
32	razmak
33	!
34	"
35	#
36	$
37	%
38	&
39	'
40	(
41	)
42	*
43	+
44	,
45	-
46	.
47	/

Dec. broj	Znak
48	0
49	1
50	2
51	3
52	4
53	5
54	6
55	7
56	8
57	9
58	:
59	;
60	<
61	=
62	>
63	?
64	@

Dec. broj	Znak
65	A
66	B
67	C
68	D
69	E
70	F
71	G
72	H
73	I
74	J
75	K
76	L
77	M
78	N
79	O

Dec. broj	Znak
80	P
81	Q
82	R
83	S
84	T
85	U
86	V
87	w
88	X
89	Y
90	Z
91	[
92	\
93	]
94	^
95	_

Tablica ASCII znakova koji se mogu ispisati

Dec. broj	Znak
96	`
97	**a**
98	b
99	c
100	d
101	e
102	f
103	g
104	h
105	i
106	j
107	k
108	l
109	m
110	n
111	o

Dec. broj	Znak
112	p
113	q
114	r
115	s
116	t
117	u
118	v
119	w
120	x
121	y
122	z
123	{
124	|
125	}
126	~
127	**DEL**

Znakovi za upravljanje ulazno-izlaznim jedinicama računala (*kontrolni znakovi*, *nonprintable*) nalaze se na pozicijama 0-31

Znakovi koji se mogu tiskati (*printable*) nalaze se na pozicijama 32-126

Na poziciji 127 nalazi se još jedan od kontrolnih znakova, znak DEL

Unicode

- Novija tablica znakova zove se Unicode (Unicode: Standard za kodiranje znakova koji je razvio Unicode Consortium.
- Korištenjem više bajtova za predstavljanje svakog znaka Unicode omogućuje da se gotovo svi pisani jezici u svijetu predstave korištenjem jednog skupa znakova

char - znakovni tip podataka

- Pohrana malih cijelih brojeva sa ili bez predznaka
- Pohrana slova, interpunkcija, posebnih znakova
- Zauzima 1 oktet.
- Definicija varijabli u programskom jeziku C:

  char [-128, 127]

  unsigned char [0, 255]
- Za prikaz se koristi međunarodni standard: ASCII kôd (American Standard Code for Information Interchange)

Znakovne konstante

`'A'`	slovo A (redni broj znaka A u ASCII tablici)
`'0'`	znamenka nula (redni broj znamenke 0)
`'\x41'`	heksadekadski zapis slova A
`'\101'`	oktalni zapis slova A
`'\a'`	simbolički zapis za znakove koji se ne mogu tiskati
`'\0'`	oznaka za kraj znakovnog niza (nul-karakter, ništični znak, znak praznoga)
`'\n'`	prijelaz u novi red
`'\\'`	znak \ ("backslash")
`'\''`	znak ' (jednostruki navodnik)
`'\"'`	znak " (dvostruki navodnik)

Primjeri sa znakovnim konstantama

- Primjer: Varijabli **c** tipa **char** pridružiti vrijednost slova A na različite načine:

  c = 'A'; pridružuje ASCII kôd znaka **'A'** koji je $\mathbf{65_{10}= 41_{16}=101_8}$

  c = 65; ili **c=0x41;** ili **c=0101;**

  ```
  c = '\x41';        /* heksadekadske konstante počinju s \x    */
  c = '\101';        /* oktalne konstante počinju s \           */
  ```

- Primjer: Varijabli **c** tipa **char** pridružiti vrijednost jednostrukog navodnika ('), a zatim pridružiti vrijednost znaka \

  ```
  c = '\'';     /* specijalni znakovi unutar navodnika
                   moraju imati ispred sebe znak \        */
  c = '\\';
  ```

- Primjer pridruživanja prilikom definicije varijable:

  ```
  char r, x='a';
  r = 'A';
  ```

Primjer: Ispisati neke znakove ASCII tablice kao broj i kao znak

```
🗁IspisAsciiZnakova
#include <stdio.h>
int main( ) {
  char x, y, p, q;
  x = 'A';
   y = x + 32;
  p = '\n';
   q = ' ';
  printf("%d %c %d %c %d %c\n", x, x, y, y, '0', '0'+1);
  printf("%d %c %d %c\n", p, p, q, q);
  return 0;
}
```

```
65 A 97 a 48 1↵
10 ↵
 32  ↵
```

Znamenke 0 do 9 u ASCII tablici

- Kada se znakovni tip koristi za pohranjivanje znamenki (0-9) treba obratiti pozornost na to da se u varijablu znakovnog tipa ne pohranjuje brojčana vrijednost te znamenke, nego ASCII vrijednost te znamenke, odnosno redni broj u ASCII tablici:

```
char a;
a = '1';
```

ekvivalentno izrazu: `a = 49;`

- U varijabli **a** nalazi se brojčana vrijednost 49, odnosno redni broj znaka `'1'` u ASCII tablici.
- Ako se želi dobiti brojčana vrijednost znamenke, potrebno je od te vrijednosti oduzeti 48. Vrijednost 48 ustvari predstavlja ASCII vrijednost znaka `'0'` .

Pretvorba ASCII znamenke u cijeli broj

- Treba uočiti da pojedine znamenke prikazane kao znak ne odgovaraju po binarnom prikazu odgovarajućem cijelom broju. Npr. znamenka 7 prikazana kao cijeli broj u 1 oktetu iznosi 0000 0111_2 , a prikazana kao ASCII znak 0011 0111_2 , odnosno 55_{10} .
- Pretvorba:

```
broj = znak - 48;
```

0011 0111 (55_{10})

```
broj = znak - '0';
```

- 8-bitni ASCII kod

  0-127 kao u 7-bitnom kodu, 128-255 prijeglasi i grafički znakovi, ovisno o tzv. kodnoj stranici

Primjeri s ASCII vrijednostima znamenaka

- Primjer:

```
char c ='A';
```

- Što će se ispisati naredbama:

```
printf ("%c", c);                    /* A */
printf ("%d", c);                    /* 65 */
printf ("%c", c + 32);               /* a */
printf ("%d", 'B' – 'A');            /* 1 */
```

Primjeri s ASCII vrijednostima znamenaka

- Primjer: Zadane su dvije varijable **a** i **b** tipa **char** koje sadrže znamenke ('0'-'9'). Napisati izraz koji će varijabli i pridružiti broj koji odgovara zbroju tih znamenki (npr. za znamenke '5' i '6' rezultat treba biti 11)

```
char a, b;
int i;
... /* pridruzivanje vrijednosti varijablama a i b */

i = a - '0' + b - '0';
```

ili

```
i = a + b – 2 * '0';
```

ili

```
i = a + b - 96;          /* 96 = 2*48 */
```

Niz znakova (string)

- Konstantni znakovni niz se označava dvostrukim navodnicima, npr.

```
"Zagreb"
```

- U memoriji se kraj niza znakova označava nul-znakom (`'\0'`). Na primjer, konstantni znakovni niz "Zagreb" u memoriji računala zauzima 7 okteta:

Z	a	g	r	e	b	\0

- primjer korištenja specijalnih znakova u konst. znakovnom nizu

```
"Znak \\ nazivamo \"backslash\""
```

- Definicija varijable u programskom jeziku C (objašnjeno kasnije):

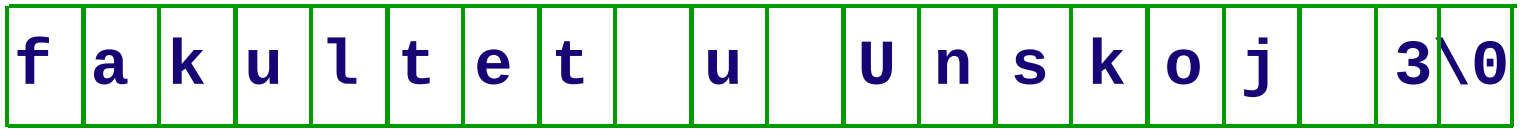
```
char ime_niza[duljina_niza+1];
```

(kao polje znakova, paziti da se rezervira mjesto za '\0')

Nastavljanje konstantnog znakovnog niza

- `"fakultet u Unskoj 3"`
- `"fakultet" " u Unskoj 3"`
- `"fakultet "  "u Unskoj 3"`
- `"fakultet"`
  `" u Unskoj 3"`

- Prethodno prikazani konstantni znakovni nizovi u memoriji računala su pohranjeni na isti način, u 20 okteta



- `"fakultet`
  `u Unskoj 3"`

Pravopisna pogreška
(dojavljuje prevodilac)

Primjeri pohrane različitih tipova konstanti

- Na koji će način prevodilac interpretirati i pretvoriti u binarni oblik sljedeće konstante:

```
 5  00000000000000000000000000000101  (cjelobrojna konstanta u 32 bita)
'5' 00110101                          (znakovna konstanta u 8 bita)
5.  0100000000010100000.....0         (realna konstanta tipa double u 64 bita)
5.f 01000000101000000...0             (realna konstanta tipa float  u 32 bita)
05  00000000000000000000000000000101 (oktalna konstanta u 32 bita)
0x5 00000000000000000000000000000101 (heksadekadska konst. u 32 bita)
"5" 00110101 00000000                 (konstantni znakovni niz 2 x 8 bita)
```

- Poseban problem studenti imaju u predočavanju raznih "nula"

```
 0   00000000000000000000000000000000 (cjelobrojna konstanta u 32 bita)
'0'  00110000                         (znakovna konstanta u 8 bita)
'\0' 00000000          (znakovna konstanta "nul-karakter", u 8 bita)
"0"  0011000000000000  (konstantni znakovni niz, 2 puta po 8 bita)
```

Uvod u programiranje

zamjena za logički tip podataka

Matematička logika: sud

- Osnovni pojam: logički sud
- Može biti istinit ili lažan
- Primjeri:

  1 < 2 je istinit

  3 > 4 je lažan
- Osnovni ili atomni sud (atom): istinitost ili lažnost utvrđuje se neposrednim zaključivanjem
- Složeniji sud tvori se formulama koje se sastoje od:

  atoma

  logičkih operatora

  zagrada

Matematička logika: osnovni operatori

- Negacija $\neg$

A	$\neg$A
1	0
0	1

- Konjunkcija $\wedge$

A	B	A$\wedge$B
0	0	0
1	0	0
0	1	0
1	1	1

- Disjunkcija $\vee$

A	B	A$\vee$B
0	0	0
1	0	1
0	1	1
1	1	1

- Ekskluzivna disjunkcija $\otimes$

A	B	A$\otimes$B
0	0	0
1	0	1
0	1	1
1	1	0

Matematička logika: korisne ekvivalencije

- S jednom varijablom i konstantama

$\neg(\neg A) = A$		**zakon dvostruke negacije**
$A \wedge A = A$		**idempotentnost konjunkcije**
$A \wedge 1 = A$		
$A \wedge 0 = 0$		
$A \wedge \neg A = 0$		
$A \vee A = A$		**idempotentnost disjunkcije**
$A \vee 1 = 1$		
$A \vee 0 = A$		
$A \vee \neg A = 1$		

Matematička logika: korisne ekvivalencije

- S dvije varijable

$A \vee B$	=	$B \vee A$	**komutativnost disjunkcije**
$A \wedge B$	=	$B \wedge A$	**komutativnost konjunkcije**
$\neg(A \wedge B)$	=	$\neg B \vee \neg A$	**de Morganov zakon**
$\neg(A \vee B)$	=	$\neg B \wedge \neg A$	**de Morganov zakon**
$A \wedge (A \vee B)$	=	A	**apsorpcija**
$A \vee (A \wedge B)$	=	A	**apsorpcija**

- S tri varijable

$A \wedge (B \wedge C)$	=	$(A \wedge B) \wedge C$	**asocijativnost konjunkcije**
$A \vee (B \vee C)$	=	$(A \vee B) \vee C$	**asocijativnost disjunkcije**
$A \wedge (B \vee C)$	=	$(A \wedge B) \vee (A \wedge C)$	**distributivnost konjunkcije**
$A \vee (B \wedge C)$	=	$(A \vee B) \wedge (A \vee C)$	**distributivnost disjunkcije**

Logička vrijednost u C-u

- U C-u ne postoji ključna riječ koja bi označavala podatak logičkog tipa, dok u nekim jezicima postoji poseban tip podataka (logical, boolean)
- Logička vrijednost odgovara vrijednosti logičkog suda (prosudbe)

DA	ili	NE
YES	ili	NO
TRUE	ili	FALSE
T	ili	F

- Svaki tip podatka u C-u je ujedno logički podatak, i to:

`istina` **ako je vrijednost** **`≠ 0`**
`laž` **ako je vrijednost** **`= 0`**

- U svrhu povećanja razumljivosti programskog koda, moguće je definirati simboličke konstante te ih u programu koristiti kao logičke konstante

```
#define TRUE 1
#define FALSE 0
...
int kisaPada = TRUE;
int sunceSija = FALSE;
```

Usporedbeni (relacijski) operatori

- Uspoređuju dva operanda. Tvore atomne sudove:

Operator	*Značenje*	*Logički izraz*	*Rezultat*
`==`	jednako	`1 == 1`	`1` (Istina)
`!=`	različito	`2 != 2 + 2`	`1` (Istina)
`>`	veće	`5 > 6`	`0` (Laž)
`>=`	veće ili jednako	`6 >= 6`	`1` (Istina)
`<`	manje	`7 < 10`	`1` (Istina)
`<=`	manje ili jednako	`7 <= 6`	`0` (Laž)

Logički operatori

- Jednostavni relacijski izrazi mogu se kombinirati u složene pomoću logičkih operatora. C uključuje 3 logička operatora:

Operator	Značenje
&&	logičko I
||	logičko ILI
!	logičko NE

- U programskom jeziku postoje i logički operatori nad bitovima, obrađeni u jednom od narednih predavanja!

Primjeri:

```
if ((x > 20) && (x < 100))
       printf("x se nalazi u otvorenom intervalu 20-100");
if ((x < 5) || (x > 20))
       printf("x se ne nalazi u zatvorenom intervalu 5-20");
if (!(x > 20))
       printf("x je manji ili jednak 20");
```

Poteškoće sa složenim logičkim uvjetima

- Složeni izrazi često se pogrešno napišu zbog "doslovnog prepisivanja" izrečenog uvjeta

- Izraz "ako je x veći od 20 i manji od 100" (uočiti da se pod "i manji od 100" podrazumijeva "i x je manji od 100"), često se "doslovno prepiše" u:

```
if ( x > 20 && < 100 )
```

što dovodi do pogreške pri prevođenju.

Ispravno:

```
if ( x > 20 && x < 100 )
```

Rezultat primjene logičkih operatora

- logička vrijednost FALSE

```
0   0.0   0.0f   0L   '\0'
```

- logička vrijednost TRUE

```
1   1.F   1.0L   0.15   148.9f -512   'a'   '\n'
```

- rezultat logičkog izraza je uvijek 0 ili 1 (tip **int**)

```
7 > 8          →  0
7.5 <= 8.5     →  1
! 1            →  0
! 15.75F       →  0
! 0            →  1
! 0.0F         →  1
! '\0'         →  1
```

Logički operatori - prioritet

	OPERATORI
	!
	< <= > >=
	== !=
	&&
	||

Primjeri:

```
if (x > 20 && x < 100)
        printf("x se nalazi u otvorenom intervalu 20-100");

if (!x > 20)
        printf("x je manji ili jednak 20");

if (!(x > 20))
        printf("x je manji ili jednak 20");
```

Logički operatori - vježba

1. Ispisati tekst "istina je" ako je učitani realni broj u intervalu [3,5] ili je u intervalu [7,9]
2. Ispisati tekst "istina je" ako je učitani cijeli broj pozitivan i ima 2 ili 4 znamenke
3. Ispisati tekst "istina je" ako uvjet iz 1. zadatka nije zadovoljen (riješiti sa i bez korištenja operatora negacije)
4. Ispisati tekst "istina je" ako uvjet iz 2. zadatka nije zadovoljen (riješiti sa i bez korištenja operatora negacije)
5. U char varijable c1 i c2 učitana su neka od velikih slova abecede (A-Z). Ispisati tekst "istina je" ako se u c1 i c2 (dakle u obje varijable) nalaze samoglasnici.
6. Ispisati tekst "istina je" ako uvjet iz 5. zadatka nije zadovoljen (riješiti sa i bez korištenja operatora negacije)