



STRUKTURE PODATAKA I ALGORITMI 1

VEŽBE 3

Danica Prodanović
Branka Andrijević
Tina Petrović



- $1 \& 1 = 1$ $0 \& 1 = 0$ $0 \& 0 = 0$
- Neka je b ili 0 ili 1. Onda $b \& 0 = 0$, $b \& 1 = b$
- **AKO ZELITE DA NEKE BITOVE POSTAVITE NA NULA, MORATE URADITI OPERACIJU AND SA 0**
- **Primer: $255 \& 15 = ?$**

255 -> 1111 1111

15 -> 0000 1111

255&15 -> 0000 1111

-> oktalno 17

-> heksadekadno 0F

-> dekadno 15



- Primer: Dato je unsigned x; Koliko je $x \& 01$?
- x binarno $B_n B_{n-1} \dots B_4 B_3 B_2 B_1 B_0$

$$\begin{array}{rcccccccc} x \& 01 = & B_n & B_{n-1} & \dots & B_4 & B_3 & B_2 & B_1 & B_0 \\ & \& 0 & 0 & \dots & 0 & 0 & 0 & 0 & 1 \\ \hline & 0 & 0 & \dots & 0 & 0 & 0 & 0 & 0 & B_0 \end{array}$$

- **ZAKLJUČAK** $x \& 01$ vraća kao rezultat nulti bit broja x



- $1 \mid 1 = 1, 1 \mid 0 = 1, 0 \mid 0 = 0$
- Ako je b ili 0 ili 1, onda $b \mid 0 = b, b \mid 1 = 1$
- **AKO ZELITE DA POSTAVITE NEKI BIT NA 1, RADITI OPERACIJU OR SA 1**
- **Primer: $255 \mid 15 = ?$**
 $255 \rightarrow 1111\ 1111$
 $15 \rightarrow 0000\ 1111$
 $255 \mid 15 \rightarrow 1111\ 1111$
 - $\rightarrow$ oktalno 377**
 - $\rightarrow$ heksadekadno FF**
 - $\rightarrow$ dekadno 255**



- Primer: Dato je unsigned x; Koliko je $x \mid 01$?
- x binarno $B_n B_{n-1} \dots B_4 B_3 B_2 B_1 B_0$

$$\begin{array}{r}
 x \mid 01 = \begin{array}{cccccccc} B_n & B_{n-1} & \dots & B_4 & B_3 & B_2 & B_1 & B_0 \\ \mid & 0 & 0 & \dots & 0 & 0 & 0 & 1 \end{array} \\
 \hline
 \begin{array}{cccccccc} B_n & B_{n-1} & \dots & B_4 & B_3 & B_2 & B_1 & 1 \end{array}
 \end{array}$$

- **ZAKLJUČAK** $x \mid 01$ vraća kao rezultat kome je 0-ti bit postavljen na 1



- Primer: Postaviti 5. bit (B_4) na 1, a ostale sacuvati

$X \quad B_n \quad B_{n-1} \quad \dots \quad B_6 \quad B_5 \quad B_4 \quad B_3 \quad B_2 \quad B_1 \quad B_0$

$| \quad 0 \quad 0 \quad \dots \quad 0 \quad 0 \quad 1 \quad 0 \quad 0 \quad 0 \quad 0 \quad (\text{dekadno } 16)$

=====

$B_n \quad B_{n-1} \quad \dots \quad B_6 \quad B_5 \quad 1 \quad B_3 \quad B_2 \quad B_1 \quad B_0$

- ZAKLJUČAK: $X | 16$ postavlja 5. bit u X na 1, a ostale bitove ne menja.
- $x | \text{pow}(2, n-1)$ postavlja n -ti bit u x na 1
- Kako postaviti 4. bit u broju x na 0, a ostale ne menjati?
- Kako postaviti najniža 3 bita u broju X na nule, a ostale bitove ne dirati?



- $1 \wedge 0 = 1, 1 \wedge 1 = 0, 0 \wedge 0 = 0$
- Neka je b ili 0 ili 1. Onda onda $b \wedge 1 = !b, b \wedge 0 = b$
- Primer: $255 \wedge 15 = ?$

$255 \rightarrow 1111\ 1111$

$15 \rightarrow 0000\ 1111$

$255 \wedge 15 \rightarrow 1111\ 0000$

$\rightarrow$ dekadno 240



- Primer: Dato je unsigned x; Koliko je $x \wedge 01$?
- x binarno $B_n B_{n-1} \dots B_4 B_3 B_2 B_1 B_0$

$$\begin{array}{cccccccc} X \wedge 01 = & B_n & B_{n-1} & \dots & B_4 & B_3 & B_2 & B_1 & B_0 \\ & \wedge & 0 & & 0 & & 0 & 0 & 1 \\ \hline & B_n & B_{n-1} & \dots & B_4 & B_3 & B_2 & B_1 & !B_0 \end{array}$$

- ZAKLJUČAK $x \wedge 01$ vraća u kome je 0-ti bit invertovan, a ostali bitovi su nepromenjeni



- Ako je $\text{SIZEOF}(\text{int}) = 1$, tj. ako se `int` registruje u 1 bajtu, tj. u 8 bitova, onda se:
1 registruje kao 0000 0001
0 registruje kao 0000 0000
- Tada je $\sim 1 = \text{INVERTOVANO}(0000\ 0001) = 1111\ 1110$ tj. 254
 $\sim 1 = (2 \text{ na } (\text{sizeof}(\text{int}) * 8)) - 2$
- Tada je $\sim 0 = \text{INVERTOVANO}(0000\ 0000) = 1111\ 1111$
 ~ 0 daje citav registar napunjen bitovima koji su 1



>> SHIFT RIGHT – POMERANJE NADESNO

- Efekat deljenja stepenom dvojke
- Na primer $32 \gg 3$ je isto sto i $32 / 2^3$,
ali $32 \gg 3$ se brže izvršava



<< SHIFT LEFT – POMERANJE ULEVO

- Efekat množenja stepenom dvojke
- Na primer $3 \ll 5$ je isto što i $3 * 32$,
ali se $3 \ll 5$ brže izvršava



- Šta je rezultat rada sledećeg programa?

```
#include <stdio.h>
main() {
    printf( "255 & 15 = %d\n", 255 & 15 );
    printf( "255 | 15 = %d\n", 255 | 15 );
    printf( "255 & 15 = %o\n", 255 & 15 );
    printf( "255 | 15 = %x\n", 255 | 15 );
    printf( "255 ^ 15 = %d\n", 255 ^ 15 );
    printf( "4 << 2    = %d\n", 4 << 2 );
    printf( "16 >> 2    = %d\n", 16 >> 2 );
    printf( "~(-3)      = %d\n", ~(-3) );
}
```



- Štampati bitove u zapisu celog broja x.

```
/* Funkcija stampa bitove datog celog broja x. Vrednost bita na
poziciji i je 0 ako i samo ako se pri konjunktiji broja x sa maskom
000..010....000 - sve 0 osim 1 na poziciji i, dobija 0. Funkcija krece
od pozicije najveće težine kreirajući masku pomeranjem jedinice u levo
za dužina(x) - 1 mesto, i zatim pomerajući ovu masku za jedno mesto u
levo u svakoj sledećoj iteraciji sve dok maska ne postane 0. */
void print_bits(int x)
{
    /* Broj bitova tipa unsigned */
    int wl = sizeof(int)*8;
    unsigned mask;
    for (mask = 1<<wl-1; mask; mask >>= 1)
        putchar(x&mask ? '1' : '0');

    putchar('\n');
}
main()
{
    print_bits(127);
    print_bits(128);
}
```



Primer ilustruje inicijalizaciju nizova.

```
#include <stdio.h>
main() {
    /* Niz inicijalizujemo tako sto mu navodimo vrednosti u
    viticasnim zagradama.
    Dimenzija niza se odredjuje na osnovu broja inicijalizatora
    */
    int a[] = {1, 2, 3, 4, 5, 6};
    /* Isto vazi i za niske karaktera */
    char s[] = {'a', 'b', 'c'};
    /* Ekvivalentno prethodnom bi bilo char s[] = {97, 98, 99}; */
    /* Broj elemenata niza */
    int a_br_elem = sizeof(a)/sizeof(int);
    int s_br_elem = sizeof(s)/sizeof(char);
    /* Ispisujemo nizove */
    int i;
    for (i = 0; i < a_br_elem; i++)
        printf("a[%d]=%d\n", i, a[i]);
    for (i = 0; i < s_br_elem; i++)
        printf("s[%d]=%c\n", i, s[i]);
}
```



- Napisati funkciju za ispis niza brojeva - demonstrira prenos nizova brojeva u funkciju.

```
#include <stdio.h>
void print_array(int a[], int n) {
    int i;
    for (i = 0; i < n; i++) { a[i]++; printf("%d ", a[i]); }
    putchar('\n');
}
main() {
    int a[] = {1, 2, 3, 4, 5, 6, 7, 8, 9};
    int i;
    printf("sizeof(a) - u okviru main : %ld\n", sizeof(a));
    int n = sizeof(a)/sizeof(int);
    print_array(a, n);
    for (i = 0; i < n; i++) printf("%d ", a[i]);
    putchar('\n');
}
```



- Napisati program koji unosi realan broj sa standardnog ulaza i ispisuje na standardni izlaz: taj broj, najmanji ceo broj ne manji ($\geq$) od tog broja, najveći ceo broj ne veći ($\leq$) od tog broja, kvadratni koren tog broja (ako broj je nenegativan), kub tog broja, kosinus tog broja.

```
#include <stdio.h>
#include <math.h>
main()
{
    double x;
    printf("Unesite broj: ");
    scanf( "%lf", &x);
    printf("\n\nUneli ste: %lf", x);
    printf("\nCeil Vaseg broja: %lf", ceil(x));
    printf("\nFloor Vaseg broja %lf", floor(x));
    if( x >= 0 )printf("\nKvadratni koren: %lf", sqrt(x) );
    else printf("\nNegativni broj" );
    printf("\nKub Vaseg broja %lf", pow(x,3));
    printf("\nCosinus: %lf\n", cos(x));
}
```

gcc -o rac zad.c -lm



- Napisati C program koji sadrži funkciju kreirajDvocifrenBroj koja za dva karaktera koja joj se prosleđuju, gde prvi argument predstavlja cifru desetice i drugi cifru jedinice, vraća ceo broj (`int kreirajDvocifrenBroj(char desetica, char jedinica)`). U glavnom delu programa se učitavaju karakteri sa standardnog ulaza (podrazumeva se da su karakteri cifre), a zatim za svaki par karaktera ispisuje karakter čija se ASCII vrednost dobija pozivom finkcije kreirajDvocifrenBroj.
- ulaz: 757932857773746932787469778532688673746946
- izlaz: KO UMIJE NJEMU DVIJE.