STRUKTURE PODATAKA I ALGORITMI 1

VEŽBE 4

Tijana Ristović
Aleksa Cerovina
Filip Radovanović
Đorđe Nedić

& BITOVSKO AND

- 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

& BITOVSKO AND

- **Primer: Dato je unsigned x; Koliko je x & 01?**
- **x binarno Bn Bn-1 ….B4 B3 B2 B1 B0**

```
x&01= Bn  Bn-1  ...  B4  B3  B2  B1  B0
    &  0   0    ...  0   0   0   0   1
=======================================
       0   0    ...  0   0   0   0   B0
```

- **ZAKLJUCAK x & 01 vraća kao rezultat nulti bit broja x**

BITOVSKO OR

- 1 | 1 = 1, 1 | 0 =1, 0 | 0 = 0
- Ako je b ili 0 ili 1, onda b | 0= b, b | 1 = 1
- AKO ZELITE DA POSTAVITE NEKI BIT NA 1, RADITI OPERACIJU OR SA 1
- Primer: 255 | 15 = ?

255 -> 1111 1111

15 -> 0000 1111

255 | 15 -> 1111 1111

-> oktalno 377

-> heksadekadno FF

-> dekadno 255

- Primer: Dato je unsigned x; Koliko je x | 01?
- x binarno Bn Bn-1 ....B4 B3 B2 B1 B0

```
x|01= Bn  Bn-1  ...  B4  B3  B2  B1  B0
    |  0   0    ...  0   0   0   0   1
==========================================
      Bn  Bn-1  ...  B4  B3  B2  B1  1
```

- ZAKLJUCAK x | 01 vraca kao rezultat kome je 0-ti bit postavljen na 1

- Primer: Postaviti 5. bit (B4) na 1, a ostale sacuvati

```
X  Bn  Bn-1  ...  B6  B5  B4  B3  B2  B1  B0
|  0   0     ...  0   0   1   0   0   0   0  (dekadno 16)
=================================================
   Bn  Bn-1  ...  B6  B5  1   B3  B2  B1  B0
```

- ZAKLJUCAK: X | 16 postavlja 5. bit u X na 1, a ostale bitove ne menja.
- `x | 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?

^ (CARET) BITOVSKO XOR – EKSKLUZIVNA DISJUNKCIJA

- **1 ^ 0 = 1, 1 ^ 1 =0, 0 ^ 0 = 0**
- **Neka je b ili 0 ili 1. Onda onda b ^ 1 = !b , b ^ 0 = b**
- **Primer: 255 ^ 15= ?**

255 -> 1111 1111

15 -> 0000 1111

255^15 -> 1111 0000

-> dekadno 240

^ (CARET) BITOVSKO XOR – EKSKLUZIVNA DISJUNKCIJA

- **Primer: Dato je unsigned x; Koliko je x ^ 01?**
- **x binarno Bn Bn-1 ….B4 B3 B2 B1 B0**

```
X^01= Bn  Bn-1  ...  B4  B3  B2  B1  B0
    ^  0   0    ...  0   0   0   0   1
=======================================
       Bn  Bn-1  ...  B4  B3  B2  B1  !B0
```

- **ZAKLJUCAK x ^ 01 vraca u kome je 0-ti bit invertovan, a ostali bitovi su nepromenjeni**

~ BITOVSKO NOT

- **Ako je SIZEOF (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 ~1 = INVERTOVANO(0000 0001) = 1111 1110 tj. 254**
  ~1 = (2 na (sizeof(int)*8)) – 2

- **Tada je ~0 = 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 >> 3 je isto sto i 32 / 2^3, ali 32 >> 3 se brže izvršava**

<< SHIFT LEFT – POMERANJE ULEVO

- **Efekat množenja stepenom dvojke**
- **Na primer 3 << 5 je isto što i 3 * 2^5,**
  ali se 3 << 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 konjunkciji broja x sa maskom
000..010....000 - sve 0 osim 1 na poziciji i, dobija 0. Funkcija krece
od pozicije najvece tezine kreirajuci masku pomeranjem jedinice u levo
za duzina(x) - 1 mesto, i zatim pomerajuci ovu masku za jedno mesto u
levo u svakoj sledecoj 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);
}
```