

ΕΠΛ232 – Προγραμματιστικές Τεχνικές και Εργαλεία

Χαμηλού Επιπέδου Προγραμματισμός (Φροντιστήριο)
Τμήμα Πληροφορικής, Πανεπιστήμιο Κύπρου

<http://www.cs.ucy.ac.cy/courses/EPL232>

Άσκηση κατανόησης 1

- Ποιο είναι το αποτέλεσμα των πιο κάτω προγραμμάτων (έστω i και j είναι μεταβλητές unsigned shorts)

```
i = 8; j = 9;  
printf("%d", i >> 1 ^ j >> 1);
```

0

```
i = 1;  
printf("%d", i & ~i);
```

0

```
i = 2; j = 1; k = 3;  
printf("%d", ~i & j ^ k);
```

2

```
i = 7; j = 8; k = 9;  
printf("%d", i ^ j & k);
```

15



Άσκηση κατανόησης 2

- Ποια είναι η επίδραση του πιο κάτω macro. Εξηγήστε,

```
#define M(x, y) ((x) ^= (y), (y) ^= (x), (x) ^= (y))
```

- Η μακροεντολή χρησιμοποιεί τον τελεστή αποκλειστικό OR για να ανταλλάξει τις τιμές των δύο ορισμάτων της, εκμεταλλευόμενη το γεγονός ότι $(a \text{ XOR } b) \text{ XOR } b$ ισούται με a . Δείτε πώς λειτουργεί η διαδικασία:

- `x` is assigned `x XOR y`
`y` is assigned `y XOR (x XOR y)`, which is `x`
`x` is assigned `(x XOR y) XOR x`, which is `y`



Άσκηση κατανόησης 2

```
#include <stdio.h>
#define M(x,y) ((x) ^=(y) , (y) ^=(x) , (x) ^=(y) )

int main() {
    unsigned int x = 4, y = 7;
    printf("%d",M(x,y));
}
```



Άσκηση κατανόησης 3

- Γράψτε τις μικροεντολές `GET_RED`, `GET_GREEN`, `GET_BLUE`, οι οποίες όταν πάρουν ένα χρώμα (`c`) σαν είσοδο, να επιστρέφουν την ένταση του χρώματος σαν 8-bit
- `#define GET_RED(c) ((unsigned char) (((c) >> 16) & 0xff))`
- `#define GET_GREEN(c) ((unsigned char) (((c) >> 8) & 0xff))`
- `#define GET_BLUE(c) ((unsigned char) ((c) & 0xff))`

Άσκηση κατανόησης 3

```
#include <stdio.h>

#define GET_RED(c)      ((unsigned char) (((c) >> 16) & 0xff))
#define GET_GREEN(c)   ((unsigned char) (((c) >> 8) & 0xff))
#define GET_BLUE(c)    ((unsigned char) ((c) & 0xff))

int main() {
    printf("%d", GET_RED(0x121416));
    printf("%d", GET_GREEN(0x121416));
    printf("%d", GET_BLUE(0x121416));
}
```



Άσκηση κατανόησης 4

- Γράψτε την ακόλουθη συνάρτηση.

`int count_ones(unsigned char ch)`
όπου θα επιστρέφει τον αριθμό από 1s bits του `ch`.

```
int count_ones(unsigned char ch)
{
    int ones = 0;
    while (ch != 0) {
        if (ch & 1)
            ones++;
        ch >>= 1;
    }
    return ones;
}
```



Άσκηση κατανόησης 5

- Γράψτε την ίδια συνάρτηση, χωρίς ωστόσο να χρησιμοποιήσετε βρόγχο.

```
int count_ones(unsigned char ch)
{
    ch = (ch & 0x55) + ((ch >> 1) & 0x55);
    ch = (ch & 0x33) + ((ch >> 2) & 0x33);
    ch = (ch & 0x0F) + ((ch >> 4) & 0x0F);
    return ch;
}
```


Άσκηση κατανόησης 6

- Γράψτε την ίδια συνάρτηση, χωρίς ωστόσο να χρησιμοποιήσετε βρόγχο.

```
int count_ones(unsigned char ch)
{
    if (ch == 0)
        return 0;
    return count_ones(ch & ch - 1) + 1;
}
```

Άσκηση κατανόησης 7

Έστω η δομή EMPLOYER με τιμές από δυφία:

```
typedef struct {
    unsigned int group:    1;    // 1 bits
    unsigned int day:      5;    // 5 bits
    unsigned int month:    4;    // 4 bits
    unsigned int year:     7;    // 7 bits
    unsigned int age:      7;    // 7 bits
    unsigned int salary:   13;   // 13 bits
} EMPLOYER;
```

Τι θα επιστρέψει η εντολή `sizeof (EMPLOYER)`, θεωρώντας ότι η μνήμη έχει 4-byte ευθυγράμμιση.

Απάντηση: 37 bits -> 8 bytes

Τι θα αλλάξει αν προσθέσουμε το `__attribute__ ((__packed__))` όρισμα;

Απάντηση: 37 bits -> 5 bytes



Άσκηση κατανόησης 8

Έστω η δομή EMPLOYER έχει τροποποιηθεί όπως πιο κάτω:

```
typedef struct {  
    unsigned int group:    1;    // 1 bits  
    unsigned int day:      5;    // 5 bits  
    unsigned int month:    4;    // 4 bits  
    unsigned int year:     7;    // 7 bits  
    unsigned int :         0;    // 0 bits  
    unsigned int salary:  13;    // 13 bits  
} __attribute__((packed)) EMPLOYER;
```

Τι θα επιστρέψει η εντολή `sizeof (EMPLOYER)`, θεωρώντας ότι η μνήμη έχει 4-byte ευθυγράμμιση. Εξηγήστε την απάντησή σας.

Απάντηση: 17 bits (ωφέλιμα) + 15 bits padding -> 4 bytes
13 bits (ωφέλιμα) + 0 bits padding -> 2 bytes
6 bytes

