



# Διάλεξη 8: Φροντιστήριο για Δομές, Δυναμική Δέσμευση Μνήμης, Αναδρομή

---

**Διδάσκων: Παναγιώτης Ανδρέου**

# ΑΠΟΡΙΕΣ

# Ερώτηση

```
penny = one  
nickel = five  
dime = ten  
quarter = twenty-five
```

Ποια από τις ακόλουθες δηλώσεις τύπου enum μπορεί να αναπαραστήσει σωστά το παράδειγμα με τα νομίσματα στα αριστερά;

**1. Επιλογή 1:**

```
enum coin {(penny,1), (nickel,5), (dime,10),  
(quarter,25)};
```

**2. Επιλογή 2:**

```
enum coin ({penny,1}, {nickel,5}, {dime,10},  
{quarter,25});
```

**3. Επιλογή 3:** 

```
enum coin  
{penny=1,nickel=5,dime=10,quarter=25};
```

**4. Επιλογή 4:**

```
enum coin  
(penny=1,nickel=5,dime=10,quarter=25);
```

**5. Επιλογή 5:**

```
enum coin {penny, nickel, dime, quarter} (1, 5, 10,  
25);
```

# Ερώτηση

```
struct Person{ //A
  char name[30];
  int id;
}

struct Person{ //B
  char name[30],
  int id;
}Person;

struct Person{ //Γ
  char name[30];
  int id;
};

typedef struct Person{ //Δ
  char name[30];
  int id;
}
```

Ποια είναι η σωστή επιλογή για να δηλώσουμε τη δομή Person;

**1. Επιλογή 1:**

A

**2. Επιλογή 2:**

B

**3. Επιλογή 3:**

Γ



**4. Επιλογή 4:**

Δ

# Ερώτηση

```
struct Person{
    char name[30];
    int id;
} Person;

int main() {
    struct Person p1;

    ??????????
}
```

Ποια είναι η σωστή επιλογή για να δώσουμε τιμή στο id του Person p1;

**1. Επιλογή 1:**

\*p1.id = 123456;

**2. Επιλογή 2:**

&p1.id = 123456;

**3. Επιλογή 3:**

p1.id = 123456;

**4. Επιλογή 4:**

p1->id = 123456;

# Ερώτηση

```
typedef struct{
    char name[30];
    int id;
} Person;

int main() {

    Person *p1;
    p1->id = 123456;

}
```

Ποια είναι η σωστή επιλογή;

**1. Επιλογή 1:**

Ο κώδικας είναι σωστός.

**2. Επιλογή 2:**

Ο κώδικας δεν κάνει compile.

**3. Επιλογή 3:** 

Ο κώδικας θα δημιουργήσει σφάλμα runtime.

**4. Επιλογή 4:**

Η μεταβλητή p1 έπρεπε να αρχικοποιηθεί με null.

# Ερώτηση

```
typedef struct {  
    char name[30];  
    int id;  
} Person;  
  
int main() {  
  
    Person *p1;  
  
    ??????????  
  
}
```

Ποια είναι η σωστή επιλογή για να αρχικοποιήσουμε το p1;

1. **Επιλογή 1:**

\*p1 = malloc(sizeof(Person));

2. **Επιλογή 2:**

p1 = (Person \*) malloc(sizeof(Person\*));

3. **Επιλογή 3:**

\*p1 = (Person \*) malloc(sizeof(Person));

4. **Επιλογή 4:** 

p1 = (Person \*) malloc(sizeof(Person));


5. **Επιλογή 5:**

&p1 = (Person \*) malloc(sizeof(Person));

# Ερώτηση

```
typedef struct {  
    char name[30];  
    int id;  
} Person;  
  
int main() {  
  
    Person * p[5];  
  
}
```

Τι κάνει ο κώδικας δίπλα;

- 1. Επιλογή 1:**  
Αρχικοποιεί ένα πίνακα με 5 στοιχεία τύπου Person
- 2. Επιλογή 2:**   
Αρχικοποιεί ένα πίνακα με 5 στοιχεία τύπου δείκτη σε δομή τύπου Person
- 3. Επιλογή 3:**  
Αρχικοποιεί 5 στοιχεία τύπου Person
- 4. Επιλογή 4:**  
Δεσμεύει μνήμη για 5 στοιχεία τύπου Person



# Ερώτηση

Ποια από τις ακόλουθες δηλώσεις δεσμεύει αρκετό χώρο για ένα πίνακα με 10 ακέραιους;

**1. Επιλογή 1:**

```
int *ptr = (int *) malloc(10, sizeof(int));
```

**2. Επιλογή 2:**

```
int *ptr = (int *) calloc(10, sizeof(int));
```

**3. Επιλογή 3:**

```
int *ptr = (int *) malloc(10*sizeof(int));
```



**4. Επιλογή 4:**

```
int *ptr = (int *) alloc(10*sizeof(int));
```

**5. Επιλογή 5:**

```
int *ptr = (int *) calloc(10*sizeof(int));
```


# Ερώτηση

```
char ptr1[] =  
    "Hello World";
```

```
char *ptr2 =  
    malloc( 5 );
```

```
ptr2 = ptr1;
```

Τι υπάρχει λάθος με τον κώδικα;  
(Υποθέτουμε ότι η malloc δεν θα αποτύχει)

1. **Επιλογή 1:**  
Θα υπάρχει διαρροή μνήμης.
2. **Επιλογή 2:**  
Δεν επιτρέπεται η ανάθεση τύπου πίνακα στον τύπο char \*.
3. **Επιλογή 3:**  
Θα υπάρξει segmentation fault.
4. **Επιλογή 4:**  
Δεν δεσμεύεται αρκετή μνήμη με τη malloc.
5. **Επιλογή 5:**   
Δεν κάνει compile

# Ερώτηση

```
typedef struct {  
    char name[30];  
    int id;  
  
    ????  
  
} Employee;
```

Πως μπορούμε να ορίσουμε για κάθε υπάλληλο ότι έχει κάποιο συγκεκριμένο manager;

**1. Επιλογή 1:**

Employee manager;

**2. Επιλογή 2:**

struct Employee manager;

**3. Επιλογή 3:**

struct \* Employee manager;

**4. Επιλογή 4:**

Employee\* manager;



# Ερώτηση

```
typedef struct {
    char name[30];

    union{
        int id;
        char passport[9];
    } eid;
} Employee[10];
```

Πως μπορούμε να έχουμε πρόσβαση στην ταυτότητα του υπάλληλου στη θέση 5;

**1. Επιλογή 1:**

`*(Employee+5).eid.id`

**2. Επιλογή 2:**

`(employees+4)->eid.id;`

**3. Επιλογή 3:**

`Employee[4].id`

**4. Επιλογή 4:**

`Employee[4].eid.id`



# Ερώτηση

```
int fibonacci (int n) {  
    return (fibonacci(n - 1) +  
    fibonacci(n - 2));  
  
    if(n==0) return 0;  
  
    if(n==1) return 1;  
}
```

Τι είναι το πρόβλημα του κώδικα στα αριστερά;

**1. Επιλογή 1:**

Δεν κάνει compile

**2. Επιλογή 2:**

Η μέθοδος δεν θα τερματίσει ποτέ!

**3. Επιλογή 3:**

Η μέθοδος δεν υπολογίζει σωστά τον αριθμό fibonacci

**4. Επιλογή 4:**

Δεν επιτρέπεται η κλήση της fibonacci εσωτερικά από την ίδια μέθοδο

# Ερώτηση

```
typedef struct {  
    char name[30];  
}  
Employee;  
  
Employee *e =  
(Employee*)  
malloc(sizeof(Employee  
));  
strcpy(e->name,  
"Panic");
```

Πως μπορούμε να επιστρέψουμε τον πρώτο χαρακτήρα του ονόματος του υπάλληλου e;

1. **Επιλογή 1:** ← e->name[0]

2. **Επιλογή 2:** ← \*(e->name)

3. **Επιλογή 3:** ← (\*e).name[0]

4. **Επιλογή 4:** ← \*((\*e).name)