Διάλεξη 17: Επανάληψη για την ενδιάμεση εξέταση

Διδάσκων: Παναγιώτης Ανδρέου

ΑΠΟΡΙΕΣ

Είναι ισοδύναμα;

```
typedef struct node {
   int          data;
   struct node *next;
} NODE;
```

A. NODE n;

B. NODE *n;

ΟΧΙ

Είναι ισοδύναμα;

```
typedef struct node {
   int          data;
   struct node *next;
} NODE;
```

A. NODE n;

B. struct node n;

ΝΑΙ

Τι χρειάζεται να προστεθεί για να είναι ισοδύναμα;

```
typedef struct node {
    int          data;
    struct node *next;
} NODE;

NODE n;
```

```
struct node {
    int          data;
    struct node *next;
};

???

NODE n;
```

A. enum NODE{node};

B. #define NODE node

C. typedef node NODE;

D. Καμία από τις επιλογές!

Ποιο είναι το σωστό;

```
typedef struct student {
    char*     name;
    int       id;
};
student students[10];
```

A. students[5]->name=“test”;

B. students[5].name=“test”; 

C. *(students+5).name=“test”;

D. Καμία από τις επιλογές!

Ποιο είναι το σωστό;

```
typedef struct student {
    char*     name;
    int       id;
};
student *s1;
```

A. *s1.name = “test”;

B. (*s1).name = “test”;

C. s1->name=“test”;

D. Καμία από τις επιλογές! 

Ποιο είναι το σωστό;

```
typedef struct student {
    char*     name;
    int       id;
};
student *s1;
```

A. s1 = malloc(sizeof(student));

B. s1 = (student*) malloc(sizeof(student)); ←

C. s1 = (student*) malloc(sizeof(student*));

D. Καμία από τις επιλογές!

Ποιο είναι το σωστό;

```
char* str = NULL;
str = (char*) malloc( sizeof( (char) ) *2 );
str = “this is a test”;
printf(“%s”, str);
```

A. this is a test 

B. this

C. th

D. Compile error

E. Segmentation Fault

Ποιο είναι το σωστό;

```
int a[] = {5, 15, 34, 54, 14, 2, 52, 72};
int *p = &a[0];
int *q = &a[2];

printf("%d", q-p);
```

A. 29

B. 2 ←

C. 8

D. Compile error

Ποιο είναι το σωστό;

```
typedef struct Student{
  char *name;  int id;
};
int main() {
     Student *s1;
     s1->id = 123456;
}
```

A. Ο κώδικας θα δημιουργήσει σφάλμα runtime. ←

B. Ο κώδικας δεν κάνει compile.

C. Η αρχικοποίηση έπρεπε να ήταν p1.id=123456;

D. Ο κώδικας είναι σωστός.

Ποιο είναι το σωστό;

```
//Θεωρείστε υλοποίηση στοίβας όπως στις διαλέξεις
STACK s;
```

<u>για να αρχικοποιήσουμε την στοίβα;</u>

A. MakeEmpty(s);

B. MakeEmpty(*s);

C. MakeEmpty(&s); ←

D. Η στοίβα είναι ήδη αρχικοποιημένη

E. Καμία από τις επιλογές!

Ποιο είναι το σωστό;

```
//Θεωρείστε υλοποίηση στοίβας όπως στις διαλέξεις
STACK *s;
```

<u>για να αρχικοποιήσουμε την στοίβα;</u>

A. MakeEmpty(s);

B. MakeEmpty(*s);

C. MakeEmpty(&s);

D. Η στοίβα είναι ήδη αρχικοποιημένη

E. Καμία από τις επιλογές! ←

Δεδομένης μίας υλοποίησης στοίβας με ΔΔΜ…

```
typedef struct node {
    int          data;
    struct node *next;
} NODE;
```

```
typedef struct stack {
    NODE  *head;
    int    size;
} STACK;
```

…το πιο κάτω είναι Σωστό ή Λάθος;

```
void Pop(STACK *S) {
    NODE *p = NULL;
    p = S->head;
    S->head = p->next;
    free(p);
    (S->size)--;
}
```

ΛΑΘΟΣ

Δεδομένης μίας υλοποίησης στοίβας με ΔΔΜ…

```
typedef struct node {
   int          data;
   struct node *next;
} NODE;
```

```
typedef struct stack {
   NODE  *head;
   int    size;
} STACK;
```

…το πιο κάτω είναι Σωστό ή Λάθος;

```
int IsEmpty(STACK *S) {
   return (S->size == 0);
}
```

ΣΩΣΤΟ

Δεδομένης μίας υλοποίησης στοίβας με ΔΔΜ…

```
typedef struct node {
    int          data;
    struct node *next;
} NODE;
```

```
typedef struct stack {
    NODE  *head;
    int    size;
} STACK;
```

…το πιο κάτω είναι Σωστό ή Λάθος;

```
void Push(STACK *S, int x){
    NODE *p = NULL;
    p->data = x;
    p->next = S->head;
    S->head = p;
    (S->size) ++;
}
```

ΛΑΘΟΣ

Μαθηματική Επαγωγή

Να αποδείξετε ότι $1+3+5+\ldots+(2n-1)=n^2$ **για n>0.**

Απόδειξη:

- Επαληθεύουμε την Π(1) ως εξής: $1=1^2$
- Υποθέτουμε ότι ισχύει η Π(k), δηλαδή ότι

$$1+3+5+\ldots+(2k-1)=k^2$$

- Και θα αποδείξουμε ότι ισχύει η Π(k+1):

$$1+3+5+\ldots+(2(k+1)-1)=(k+1)^2$$

$$1+3+5+\ldots+(2k-1)+(2(k+1)-1)$$

$$=[1+3+5+\ldots+(2k-1)]+(2k+1)$$

$$=k^2+2k+1 \quad \text{[Επαγωγική υπόθεση]}$$

$$=(k+1)^2 \quad \textbf{[Το οποίο είναι το ζητούμενο]}$$

Επανάληψη Χρήσιμων Μαθηματικών Ορισμών

Ορισμός 1: $\log_x a = b$ **iff** $x^b = a$

Χρήσιμοι νόμοι λογάριθμων:

$\log ab = \log a + \log b$ $\quad\quad$ $\log a^b = b.\log a$

$\log a \div b = \log a - \log b$ $\quad\quad$ $\log_a b = (\log_c b) \div (\log_c a)$

$b^{\log_b x} = x$ $\quad\quad$ $\log a = 1$

$\log^2 n = (\log n)^2 = \log n * \log n$ $\quad\quad$ $\log n^2 = \log n + \log n = 2x\log n$

$\log_{10}$ (common) , $\log_e$ (ln - natural), **$\log_2$ (lg - binary)**

π.χ., $\log_2 2=1$ $\quad$ $\log_2 1=0$ $\quad\quad$ $\log_2 0$ δεν ορίζεται

Ορισμός 2: $\lfloor x \rfloor = \max\{a \mid a \le x, \text{int}(a)\}$ (floor)

π.χ., $\lceil x \rceil = \min\{a \mid a \ge x, \text{int}(a)\}$ (ceiling)

$$\left\lceil \frac{63}{11} \right\rceil = \lceil 5.72 \rceil = 6, \quad \lfloor 5.1634 \rfloor = 5$$

Ακολουθίες και Αθροίσματα

$$\sum_{i=0}^{n} i = \frac{n \cdot (n+1)}{2}$$

$$\sum_{i=0}^{n} i^2 = \frac{n \cdot (n+1)(2n+1)}{6}$$

$$\sum_{i=0}^{n} 2^i = 2^{n+1} - 1$$

$$\sum_{i=0}^{n} a^i = \frac{a^{n+1} - 1}{a - 1}$$

Άθροισμα Γεωμετρικής Προόδου

$$\sum_{i=0}^{\log_2 n} 2^i = n + n/2 + n/4 + \ldots + 2 + 1 = 2n - 1$$

Το οποίο προκύπτει αλλιώς ως: $\sum_{i=0}^{\log_2 n} 2^i = \frac{2^{\log_2 n+1}-1}{2-1} = 2^{\log_2 n} * 2^1 - 1 = 2n - 1$

Παράδειγμα: Ανάλυση Χρόνου Εκτέλεσης

```
void mystery1(){
    …
    for (i=0; i<lgn; i++){
        for (j=0;j<i²;j++){
            sum++;
        }                    Εσωτερικός Βρόγχος
    }                        Εξωτερικός Βρόγχος
}
```

Ανάλυση

- **Εσωτερικός Βρόγχος:** $IL = \sum_{j=0}^{i^2} 1 = i^2$
- **Εξωτερικός Βρόγχος:**

$$OL = \sum_{i=0}^{\lg n} IL = \sum_{i=0}^{\lg n} i^2 = \frac{\lg n\,(\lg n + 1)(2\lg n + 1)}{6} \in \Theta\left(\lg^3 n\right)$$

Παράδειγμα: Ανάλυση Χρόνου Εκτέλεσης

```
void mystery2(){
    …
    for (i=0; i<n; i++){
        for (j=0;j<n*i;j++){
            sum++;
        }
    }
}
```

Εσωτερικός Βρόγχος

Εξωτερικός Βρόγχος

Ανάλυση

- **Εσωτερικός Βρόγχος:** $IL = \sum_{j=1}^{n*i} 1 = n \times i$
- **Εξωτερικός Βρόγχος:**

$$OL = \sum_{i=0}^{n} IL = \sum_{i=0}^{n} n \times i = n \sum_{i=0}^{n} i = n \frac{n(n+1)}{2} \in \Theta(n^3)$$

Παράδειγμα: Ανάλυση Χρόνου Εκτέλεσης

```
void mystery3(){
    …
    for (i=0; i<n; i++){
        for (j=n*i;j>0;j--){
            sum++;
        }
    }
}
```

Εσωτερικός Βρόγχος

Εξωτερικός Βρόγχος

Ανάλυση

- **Εσωτερικός Βρόγχος:**

$$IL = \sum_{j=1}^{n*i} 1 = n \times i$$

- **Εξωτερικός Βρόγχος:**

$$OL = \sum_{i=0}^{n} IL = \sum_{i=0}^{n} n \times i = n \sum_{i=0}^{n} i = n \frac{n(n+1)}{2} \in \Theta(n^3)$$

Παράδειγμα: Αντικατάσταση

```
void mystery4(int n){
   …
   for (i=0; i<n; i++){ sum++; }

   if(n>1) return mystery4(n/2) +
                  mystery4(n/2);
   else    return 1;
}
```

Μορφή

$$T(n) = n + 2xT(n/2)$$

Παράδειγμα: Αντικατάσταση

Έχουμε την αναδρομική εξίσωση

$T(n) = 2 \cdot T(n/2) + n$, για κάθε $n>1$

$T(1) = 1$

Τότε, αντικαθιστώντας το $T(n/2)$ με την τιμή του παίρνουμε

$T(n) = 2 \cdot T(n/2) + n$ // Εκτέλεση 1

$= 2(2 \cdot T(n/4) + n/2) + n$ // Εκτέλεση 2

$= 2^2 \cdot T(n/4) + n + n$ // Πράξεις

$= 2^3 \cdot T(n/8) + n + n + n$ // Εκτέλεση 3

$= 2^k \cdot T(n/2^k) + k.n$ // $k=\log_2 n$ ➔ $2^k=n$

$= 2^k \cdot T(1) + k.n$ // $n=2^k$, $T(n/2^k)=T(1)=1$

$= 2^{\log n} + \log_2 n.n$ // Πράξεις

$= n + n\log_2 n$ // $n=2^k$, $T(n/2^k)=T(1)=1$

$\in \mathbf{\Theta(n\log_2 n)}$

Παράδειγμα: Αντικατάσταση

T(n) = 2xT(n/2) + n

Master Theorem: Έστω ότι η f είναι μία αύξουσα που ικανοποιεί τη λειτουργία της επανάληψης:

$$f(n) = af(n/b) + cn^d$$

όταν το $n = b^k$, όπου k είναι ένας θετικός ακέραιος, $a \geq 1$, b > 1, και c, d είναι πραγματικοί αριθμοί, $c \geq 0, d \geq 0$. Τότε ισχύει:

$$f(n) is \begin{cases} O(n^d) & if\ a < b^d \\ O(n^d log(n)) & if\ a = b^d \\ O(n^{log_b(a)}) & if\ a > b^d \end{cases}$$

a=2, b=2, c=1, d=1 $\quad$ **$b^d = 2^1 = 2$ → $a = b^d$**

$$f(n) is \begin{cases} O(n^d log(n)) & if\ a = b^d \end{cases}$$

$T(n) \in O(n^1 log(n))$ → $T(n) \in O(nlog(n))$