



Day 1

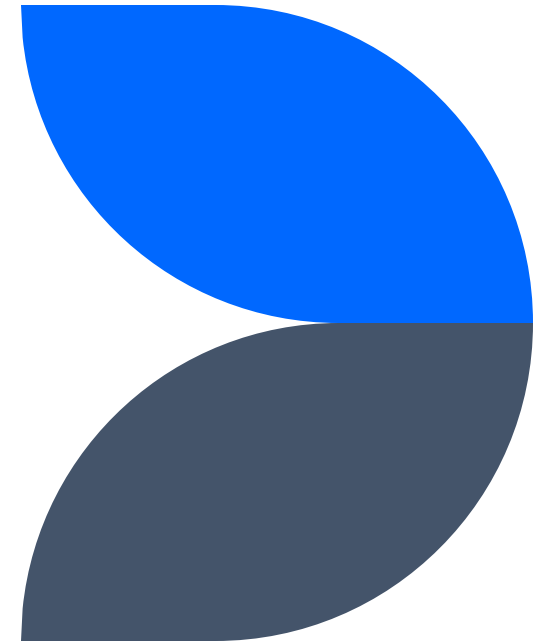
Juniors Summer Camp 2023



Πρόγραμμα

- Data Types
- If statements
- For loops
- Arrays / Strings
- Μνήμη / Χρόνο
- Functions
- Struct
- Problem Solving

Data types



Data types

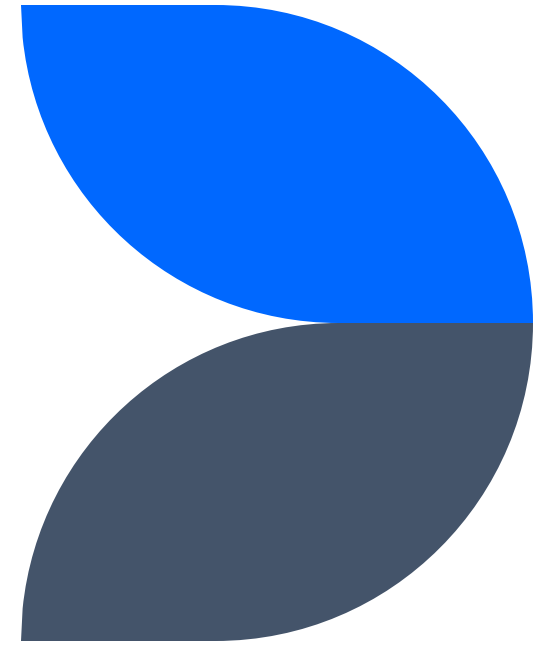
- int : -2147483648 to 2147483647
- long int : -9223372036854775808 to 9223372036854775807
- long long int : $-(2^{63})$ to $(2^{63})-1$
- unsigned long long int : 0 to 18,446,744,073,709,551,615

- float : δεκαδικούς (32 bits)
- double : δεκαδικούς (64 bits)

- char : χαρακτήρες
- string : συμβολοσειρά

- bool : true or false

Modulo operator



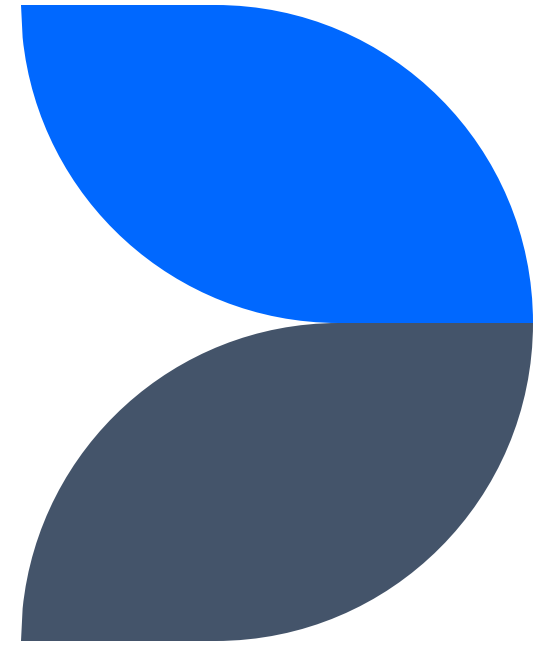
Modulo operator (%)

- **Μόνο** μεταξύ ακεραίων
 - Δίνει το υπόλοιπο της διαίρεσης
 - Σύμβολο στην c++ : %
-
- Παράδειγμα: $5 \bmod 3 = 2$
 - Παράδειγμα: $7 \bmod 9 = 7$

Οι βασικές πράξεις

Operator	Result	Operator	Result
+	Addition	+=	Addition and assignment
-	Subtraction	-=	Subtraction and assignment
*	Multiplication	*=	Multiplication and assignment
/	Division	/=	Division and assignment
%	Mod Division	%=	Mod division and assignment
++	Increment	--	Decrement

If statements

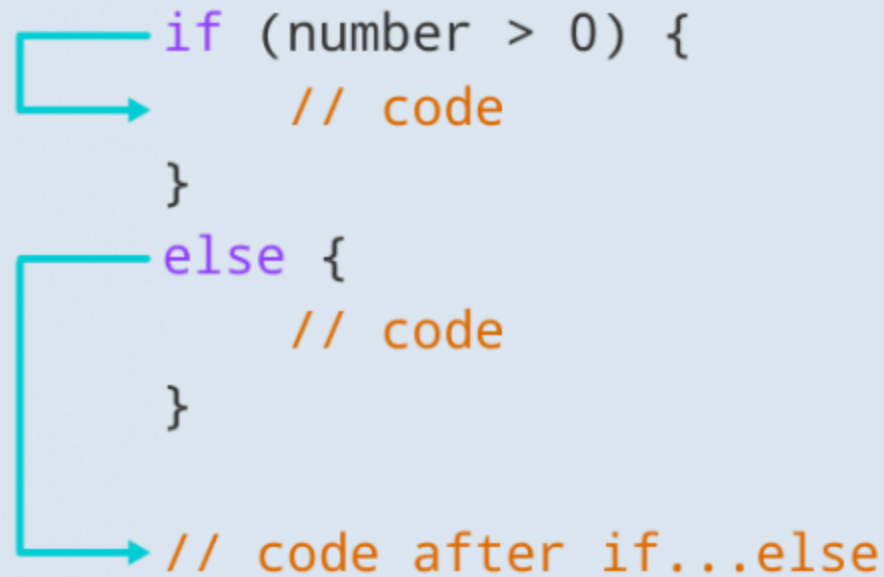


If statements

Condition is true

```
int number = 5;
```

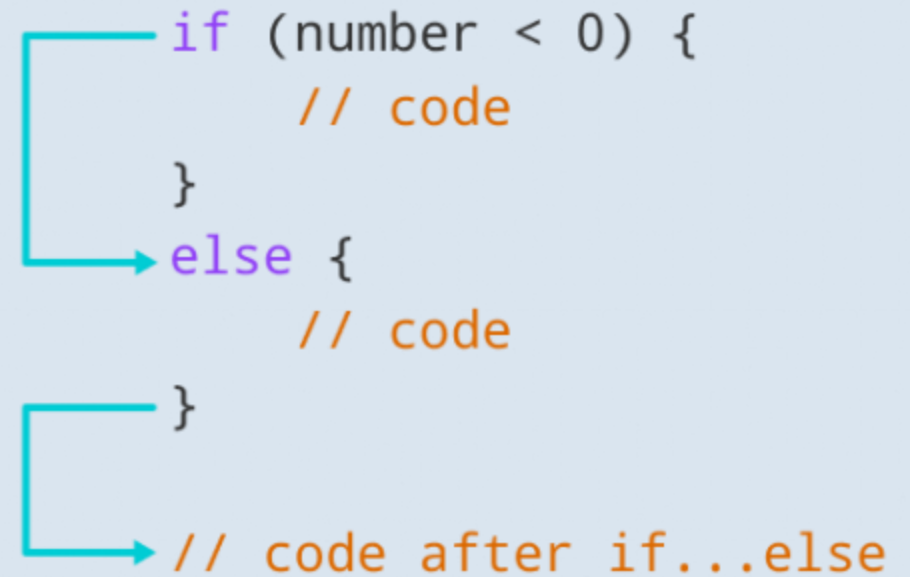
```
if (number > 0) {  
    // code  
}  
else {  
    // code  
}  
// code after if...else
```



Condition is false

```
int number = 5;
```

```
if (number < 0) {  
    // code  
}  
else {  
    // code  
}  
// code after if...else
```



Δίσεκτο

Problem

Submissions

Leaderboard

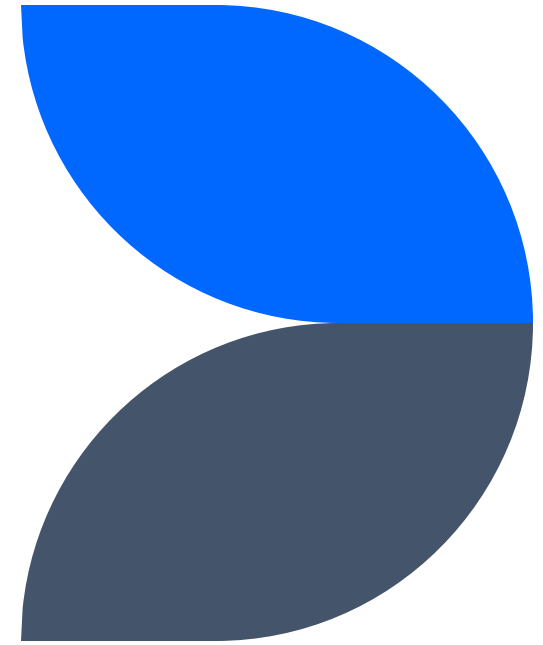
Discussions

Να δημιουργήσετε πρόγραμμα, το οποίο να δέχεται ένα ακέραιο αριθμό που αντιστοιχεί σε μια χρονολογία και να βρίσκει αν το έτος είναι δίσεκτο. Ένα έτος είναι δίσεκτο αν διαιρείται ακριβώς με το 4, εκτός αν διαιρείται ακριβώς και με το 100 οπότε και δεν είναι δίσεκτο. Ωστόσο, αν διαιρείται με το 400 τότε το έτος είναι δίσεκτο. Για παράδειγμα, το 2012 είναι δίσεκτο γιατί $2012/4 = 503$, όπως και το 1200 είναι δίσεκτο γιατί $1200/400 = 3$. Το 2100 δεν είναι δίσεκτο γιατί $2100/100 = 21$.

<https://www.hackerrank.com/contests/control-structures/challenges/challenge-141>

```
1 #include <cmath>
2 #include <cstdio>
3 #include <vector>
4 #include <iostream>
5 #include <algorithm>
6 using namespace std;
7
8
9 int main() {
10
11     int year;
12     cin>>year;
13
14     if(year%400==0){
15         cout<<1<<endl; // einai disektos
16     }
17     else
18     if(year%4 ==0 && year%100!=0){
19         cout<<1<<endl; // einai disektos
20     }
21     else
22         cout<<0<<endl; // den einai disektos
23
24     return 0;
25 }
```

For Loops



For – While

- For(int i=0 ; i<n ; i++){

//κώδικας

}

- While(συνθήκη){

//κώδικας

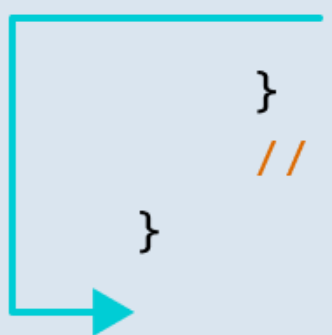
}

- Πότε χρησιμοποιώ while και πότε χρησιμοποιώ for;

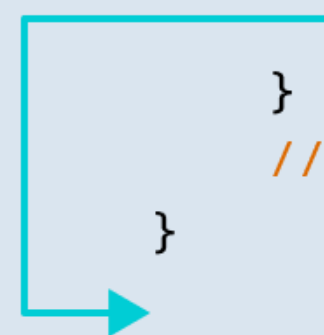
break command

- Σταματά και βγαίνει έξω από τον βρόχο

```
for (init; condition; update) {  
    // code  
    if (condition to break) {  
        break;  
    }  
    // code  
}
```



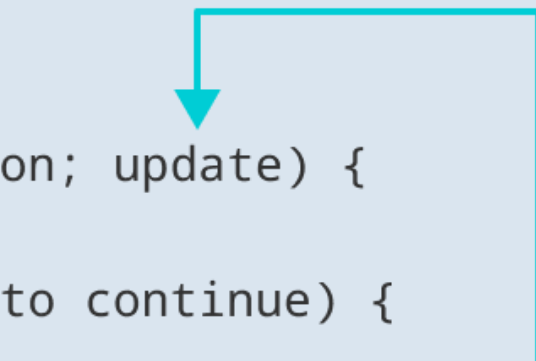
```
while (condition) {  
    // code  
    if (condition to break) {  
        break;  
    }  
    // code  
}
```



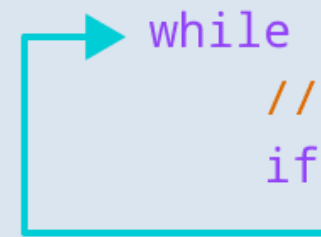
continue command

- Ξεκινάει τον βρόχο από την αρχή (με την επόμενη επανάληψη)

```
for (init; condition; update) {  
    // code  
    if (condition to continue) {  
        continue;  
    }  
    // code  
}
```

A cyan arrow starts from the 'continue;' statement, moves right, then up, then left, and finally down to point at the beginning of the 'for' loop's opening curly brace, illustrating that the loop restarts from the beginning.

```
while (condition) {  
    // code  
    if (condition to continue) {  
        continue;  
    }  
    // code  
}
```

A cyan arrow starts from the 'continue;' statement, moves left, then up, and finally right to point at the beginning of the 'while' loop's opening curly brace, illustrating that the loop restarts from the beginning.

Άθροισμα πεδίου τιμών

Problem

Submissions

Leaderboard

Discussions

Αν σας δοθούν δύο ακέραιοι αριθμοί A , B ($A < B$), να βρείτε το άθροισμα όλων των ακεραίων στο διάστημα A μέχρι B , συμπεριλαμβανομένων.

<https://www.hackerrank.com/contests/looping-around/challenges/challenge-1642/problem>


```
1 #include<iostream>
2 using namespace std;
3 int main () {
4
5     int a,b;
6     cin>>a>>b;
7
8     int s=0; // arxikopoiisi
9
10    for(int i=a;i<=b;i++){
11        s+=i;
12    }
13
14    cout<<s<<endl;
15
16    return 0;
17 }
```

- Υπάρχει πιο γρήγορος τρόπος υπολογισμού αυτού του αθροίσματος;

```
1 #include<iostream>
2 using namespace std;
3 int main () {
4
5     int a,b;
6     cin>>a>>b;
7
8     int n=b-a+1;
9     int s=n*(a+b)/2;
10
11     cout<<s<<endl;
12
13     return 0;
14 }
```

- Γιατί είναι πιο γρήγορο; (day 2 πολυπλοκότητες)

Ψηφία αριθμού

Problem

Submissions

Leaderboard

Discussions

Αν σας δοθεί ένας ακέραιος αριθμός N στο διάστημα $[1 \dots 1,000,000,000]$, να δημιουργήσετε ένα πρόγραμμα το οποίο να επιστρέφει το πλήθος των ψηφίων του.

<https://www.hackerrank.com/contests/looping-around/challenges/challenge-1642/problem>

- Επίσης να υπολογιστεί το άθροισμα των ψηφίων του αριθμού

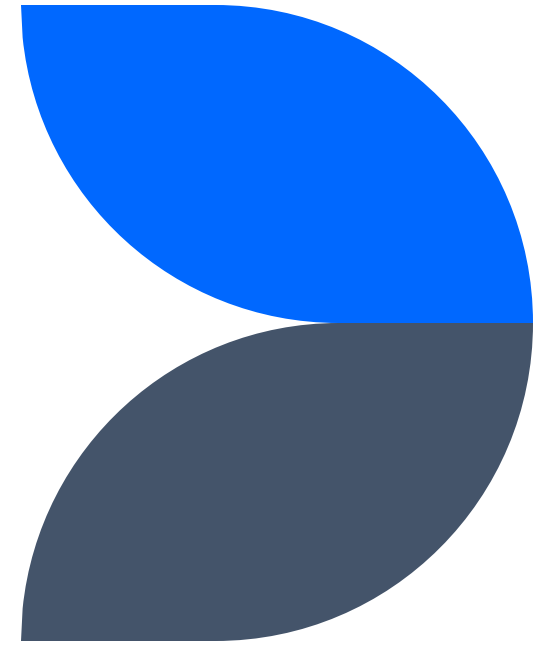
```

1 #include <cmath>
2 #include <cstdio>
3 #include <vector>
4 #include <iostream>
5 #include <algorithm>
6 using namespace std;
7
8
9 int main() {
10
11     int a;
12     cin>>a;
13
14     int c=0; // plithos psifion
15     int s=0; // athroisma psifion
16
17     while(a>0){
18         s=s+a%10;
19         a/=10;
20         c++;
21     }
22
23     cout<<c<<endl;
24     cout<<s<<endl;
25
26     return 0;

```

Πίνακες

Strings



Πίνακες

- Με τους πίνακες μπορούμε να ομαδοποιήσουμε δεδομένα του ίδιου τύπου για την πιο εύκολη διαχείριση τους
- Δήλωση : `type name[size];` (πχ. `int a[100];`)
- `int a[n]={};` //αρχικοποίηση πίνακα με όλα τα στοιχεία 0

Elements	2	7	4	8	1	6
Index	0	1	2	3	4	5

String

- Το string είναι ένας πίνακας από χαρακτήρες (char)

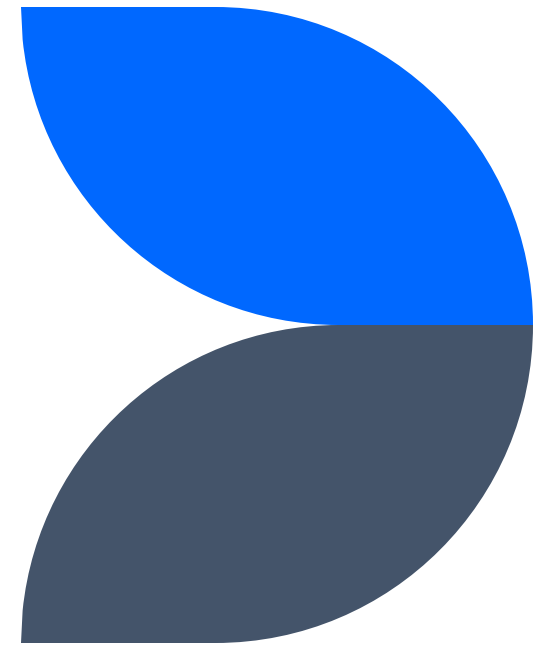
```
string a="ejoi";
```

```
cout<<a.size()<<endl; // Μέγεθος string: 4
```

```
cout<<a[2]<<endl; // ο (ξεκινάει η αρίθμηση από το 0)
```

```
getline(cin,b); //διαβάζει όλη την γραμμή και τα κενά
```

Συναρτήσεις



Συναρτήσεις

- Οι συναρτήσεις είναι χρήσιμες για την επαναχρησιμοποίηση κώδικα και για την γενικότερη οργάνωση του προγράμματος μας
- Μία συνάρτηση μπορεί να μη επιστρέφει τίποτα (void) ή να επιστρέφει οπουδήποτε τύπο δεδομένων θέλουμε (πχ int)

- Να δημιουργήσετε μία συνάρτηση που βρίσκει τον μέγιστο κοινό διαιρέτη και τον επιστρέφει

```
1  #include <iostream>
2  using namespace std;
3
4  int gcd(int a, int b) {
5
6      int gcd = 1;
7
8      for (int i = 1; i <= min(a,b); i++) {
9          if (a % i == 0 && b % i == 0) {
10             gcd = i;
11         }
12     }
13
14     return gcd;
15 }
16
17 int main() {
18
19     int a,b;
20     cin>>a>>b;
21     cout<<gcd(a,b)<<endl;
22
23 }
```

Αναδρομή

- Μία συνάρτηση όταν καλεί τον εαυτό της λέγεται αναδρομική συνάρτηση
- Σε μία αναδρομική συνάρτηση πρέπει να βάζουμε κάποια συνθήκη επιστροφής
- Να κάνετε μια αναδρομική συνάρτηση που υπολογίζει το παραγοντικό
- Να κάνετε μια αναδρομική συνάρτηση που υπολογίζει τον k όρο της ακολουθίας fibonacci

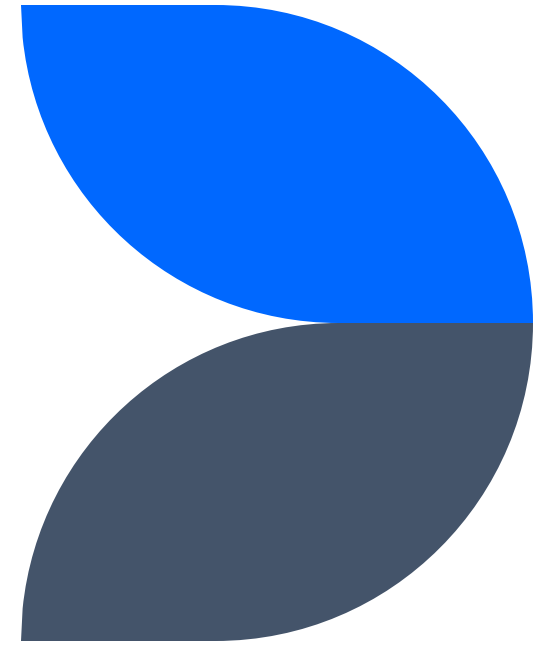
Παραγοντικό

```
int paragontiko(int n) {  
    if (n==0)  
        return 1;  
    return n * paragontiko(n-1);  
}
```

Fibonacci

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

Struct



Struct

- Με το struct μπορούμε να δημιουργήσουμε δικούς μας τύπους δεδομένων που να περιέχουν διάφορους τύπους

```
struct name{  
type1 name;  
type2 name;  
};
```

Παράδειγμα

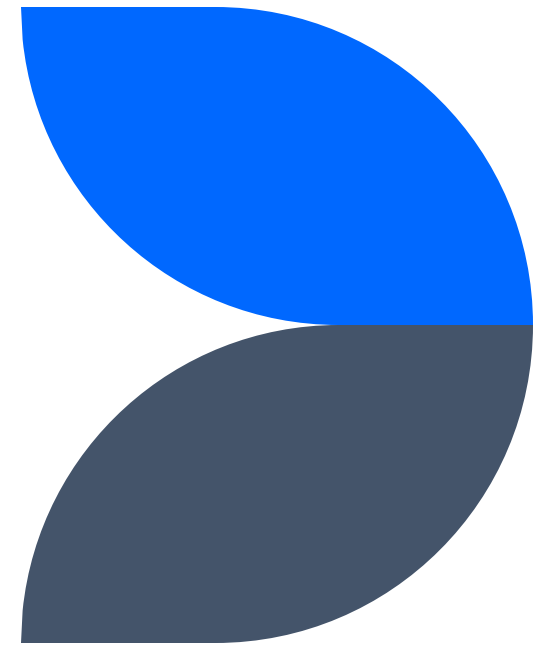
```
struct mathitis{
    string name;
    int vathmos;
};

int main(){
    mathitis a[10];

    for(int i=0;i<10;i++){
        cin>>a[i].name>>a[i].vathmos;
    }

    cout<<a[5].name<<" "<<a[5].vathmos<<endl;
}
```

Χρόνος - Μνήμη



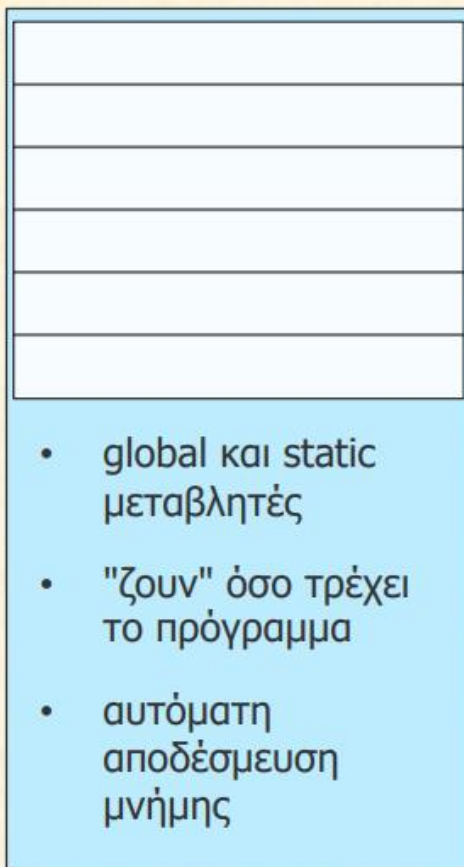
Μνήμη

- Κάθε τύπος δεδομένου έχει καταλαμβάνει διαφορετικό χώρο στην μνήμη
- Συνήθως η μνήμη δεν αποτελεί πρόβλημα στον ανταγωνιστικό προγραμματισμό.
- (Όταν θέλω να δηλώσω ένα μεγάλο πίνακα καλύτερα να τον δηλώνω global επειδή επιτρέπει την μεγαλύτερη δέσμευση μνήμης)

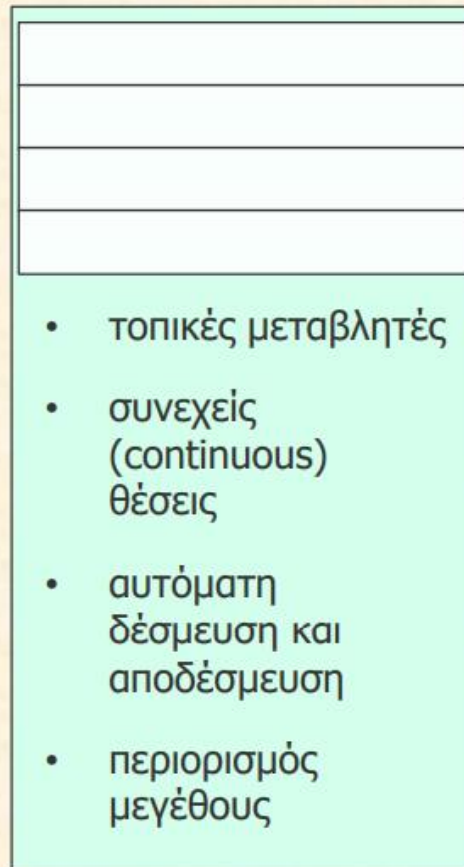
Type	Bits	Range
int	16	-32768 to 32767
unsigned int	16	0 to 65535
signed int	16	-31768 to 32767
short int	16	-31768 to 32767
unsigned short int	16	0 to 65535
signed short int	16	-32768 to 32767
long int	32	-2147483648 to 2147483647
unsigned long int	32	0 to 4294967295
signed long int	32	-2147483648 to 2147483647
float	32	3.4E-38 to 3.4E+38
double	64	1.7E-308 to 1.7E+308
long double	80	3.4E-4932 to 3.4E+4932
char	8	-128 to 127
unsigned char	8	0 to 255
signed char	8	-128 to 127

Δέσμευση μνήμης

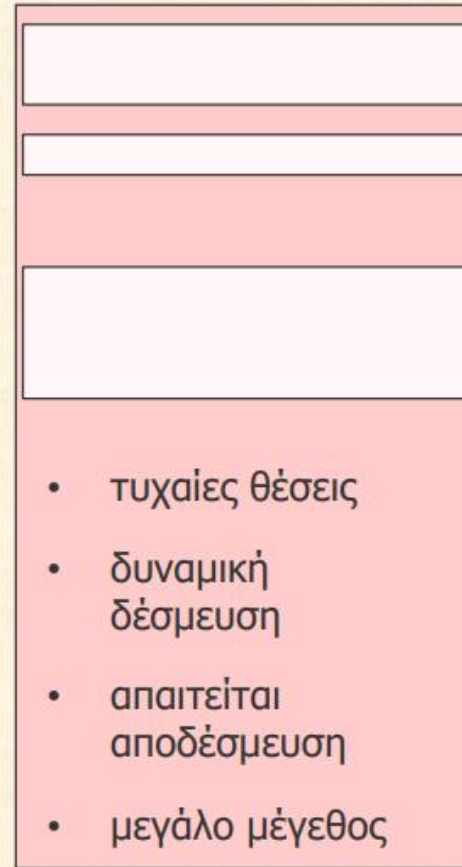
static



stack



heap



Δέσμευση μνήμης

◆ Static

- global ή static μεταβλητές

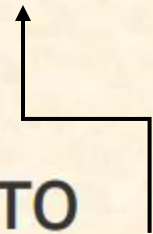
◆ Stack

- μεταβλητές που δηλώνονται μέσα σε συναρτήσεις

◆ Heap

- μεταβλητές με δυναμική δέσμευση
- αποδέσμευση από τον προγραμματιστή ή το λειτουργικό, με το πέρας του προγράμματος

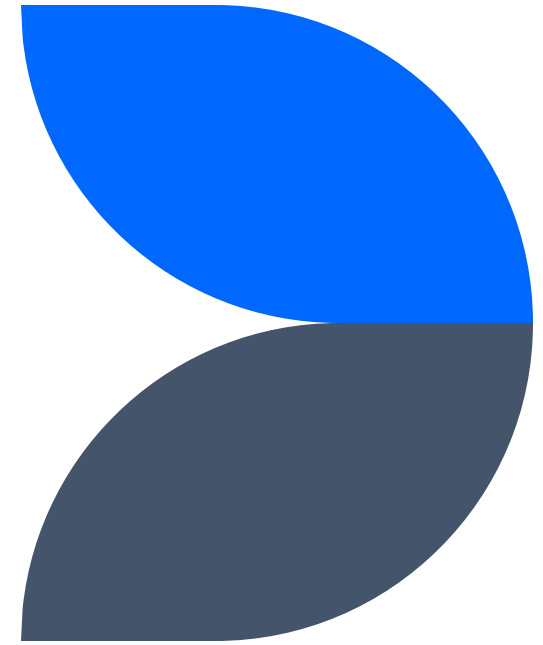
```
int a* = new int[n];  
delete [] a;
```



Χρόνος

- Πολλές φορές ενώ το πρόγραμμα μας βγάζει σωστά αποτελέσματα δεν παίρνουμε **όλους** του πόντους επειδή είναι «αργό»
- Περίπου μπορεί να γίνουν 10,000,000 πράξεις το δευτερόλεπτο
- Αν ξεπεράσουμε το χρονικό όριο του προβλήματος, τότε παίρνουμε το error TLE (time limit exceeded) , που είναι πολύ συνηθισμένο
- (Day 2 πολυπλοκότητες)

Problem Solving



Ασκήσεις

(easy if - for)

- <https://www.hackerrank.com/contests/easy-arrays/challenges/challenge-1684>

(medium string)

- <https://www.hackerrank.com/contests/stringers/challenges/challenge-207>

(medium array)

- <https://www.hackerrank.com/contests/easy-arrays/challenges/challenge-3162>



Τέλος Day 1

Juniors Summer Camp 2023