

# Διάλεξη 11 - Δεδομένα Εισόδου

Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών

Εισαγωγή στον Προγραμματισμό

Θανάσης Αυγερινός

## Ανακοινώσεις / Διευκρινήσεις

- Δεν έχουμε διάλεξη ή εργαστήρια την Παρασκευή, 17 Νοεμβρίου

# Την προηγούμενη φορά

- Δείκτες και Πίνακες reloaded
  - Δισδιάστατοι πίνακες
  - Παραδείγματα
  - Δείκτες σε δείκτες
  - Δυναμική διαχείριση μνήμης

# Σήμερα

- Δεδομένα Εισόδου (και εξόδου)
- ~~Μνήμη~~ (next time)
- Παραδείγματα

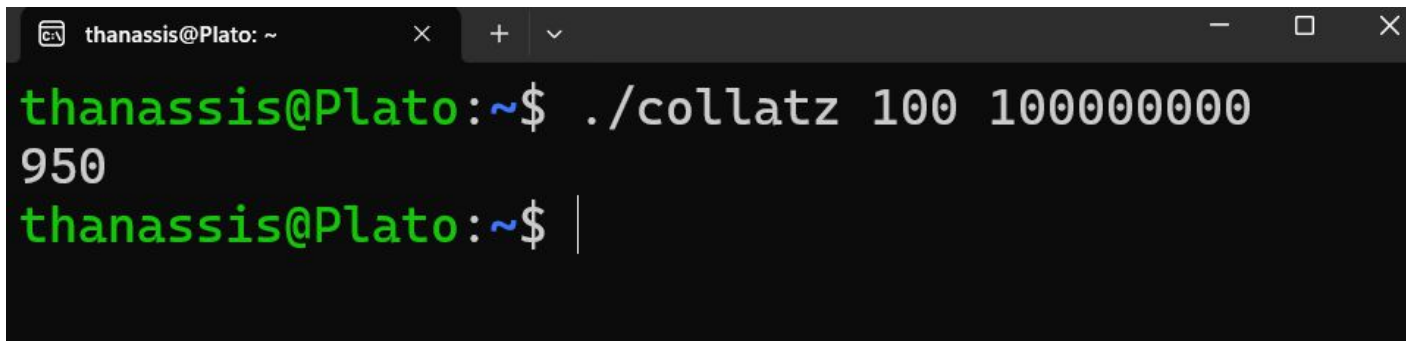
# Δεδομένα Εισόδου και Εξόδου (Input and Output Data)



## Δεδομένα Εισόδου (Input Data)

Τα **δεδομένα εισόδου** (**input data**) είναι μια σειρά από χαρακτήρες (bytes) τα οποία ο χρήστης δίνει στο πρόγραμμα. Υπάρχουν 4 μέθοδοι να εισάγουμε δεδομένα:

1. **Ορίσματα** στην γραμμή εντολών

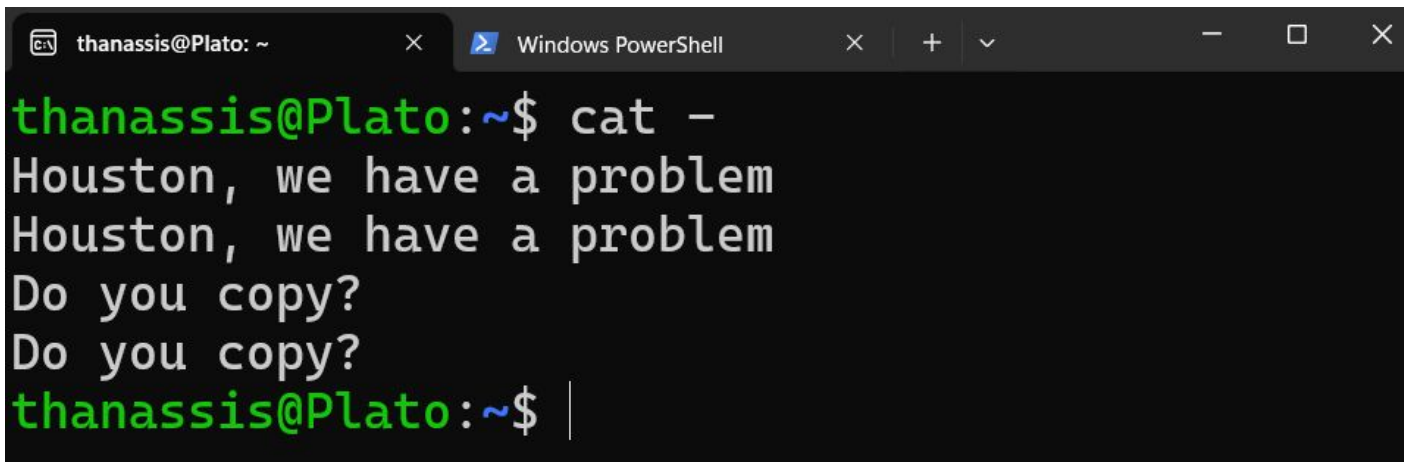


```
thanassis@Plato: ~  
thanassis@Plato:~$ ./collatz 100 100000000  
950  
thanassis@Plato:~$ |
```

## Δεδομένα Εισόδου (Input Data) - 2/4

Τα **δεδομένα εισόδου** (**input data**) είναι μια σειρά από χαρακτήρες (bytes) τα οποία ο χρήστης δίνει στο πρόγραμμα. Υπάρχουν 4 μέθοδοι να εισάγουμε δεδομένα:

2. Γράφοντας κείμενο στην **πρότυπη είσοδο** (**standard input** ή **stdin**) συνήθως με το πληκτρολόγιο

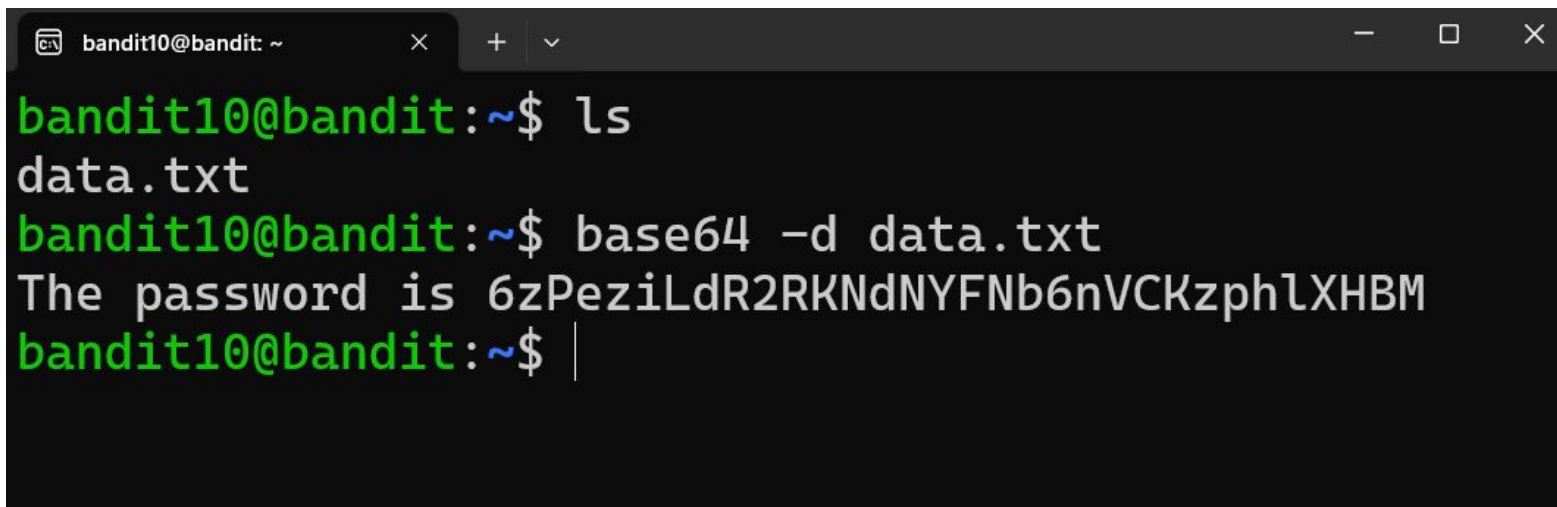


```
thanassis@Plato: ~  
Windows PowerShell  
thanassis@Plato:~$ cat -  
Houston, we have a problem  
Houston, we have a problem  
Do you copy?  
Do you copy?  
thanassis@Plato:~$ |
```

## Δεδομένα Εισόδου (Input Data) - 3/4

Τα **δεδομένα εισόδου** (input data) είναι μια σειρά από χαρακτήρες (bytes) τα οποία ο χρήστης δίνει στο πρόγραμμα. Υπάρχουν 4 μέθοδοι να εισάγουμε δεδομένα:

3. Διαβάζοντας **αρχεία** από το σύστημα αρχείων (επόμενες διαλέξεις)



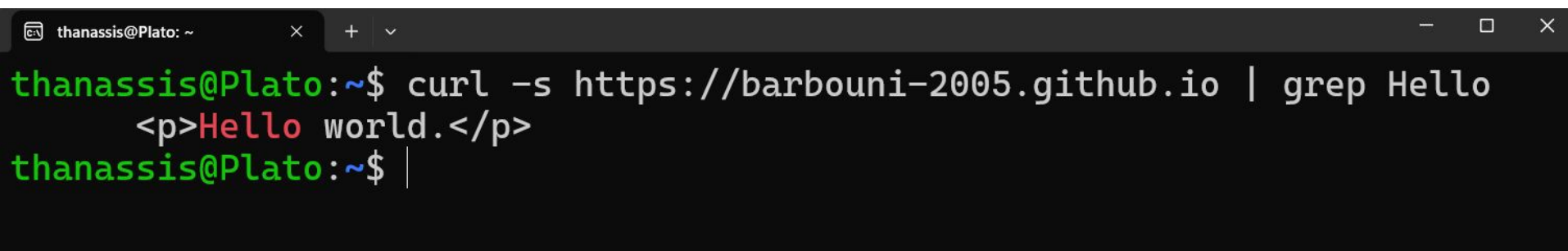
```
bandit10@bandit: ~  
bandit10@bandit:~$ ls  
data.txt  
bandit10@bandit:~$ base64 -d data.txt  
The password is 6zPezilDR2RKNdNYFNb6nVCKzphlXHBM  
bandit10@bandit:~$ |
```



## Δεδομένα Εισόδου (Input Data) - 4/4

Τα **δεδομένα εισόδου** (**input data**) είναι μια σειρά από χαρακτήρες (bytes) τα οποία ο χρήστης δίνει στο πρόγραμμα. Υπάρχουν 4 μέθοδοι να εισάγουμε δεδομένα:

4. Διαβάζοντας από το **δίκτυο** ή άλλες πηγές - π.χ., User Interface (σε επόμενα εξάμηνα)



```
thanassis@Plato: ~  
thanassis@Plato:~$ curl -s https://barbouni-2005.github.io | grep Hello  
<p>Hello world.</p>  
thanassis@Plato:~$
```

## Δεδομένα Εισόδου (Input Data)

Τα **δεδομένα εισόδου** (**input data**) είναι μια σειρά από χαρακτήρες (bytes) τα οποία ο χρήστης δίνει στο πρόγραμμα. Υπάρχουν 4 μέθοδοι να εισάγουμε δεδομένα:

1. **Ορίσματα** στην γραμμή εντολών ✓
2. Γράφοντας κείμενο στην **πρότυπη είσοδο** (**standard input** ή **stdin**) ←
3. Διαβάζοντας **αρχεία** από το σύστημα αρχείων (επόμενες διαλέξεις)
4. Διαβάζοντας από το **δίκτυο** ή άλλες πηγές (άλλα εξάμηνα)

25%



## Η συνάρτηση `getchar()`

Η συνάρτηση `getchar()` ορίζεται στο `stdio.h`, διαβάζει έναν χαρακτήρα εάν υπάρχει από το `stdin` του προγράμματος και τον επιστρέφει ως ακέραιο. Αν δεν υπάρχει, επιστρέφει την τιμή End-Of-File / EOF (-1).

Η συνάρτηση `getchar()` είναι "έτοιμη" για χρήση από μας από το `stdio.h` - πως μπορώ να βρω πως συμπεριφέρεται;

Ανοίγω ένα τερματικό και τρέχω `man  
getchar!`

## Η συνάρτηση `getchar()`

Η συνάρτηση `getchar()` ορίζεται στο `stdio.h`, διαβάζει έναν χαρακτήρα εάν υπάρχει από το `stdin` του προγράμματος και τον επιστρέφει ως ακέραιο. Αν δεν υπάρχει, επιστρέφει την τιμή End-Of-File / EOF (-1). Η συνάρτηση έχει την ακόλουθη μορφή:

```
int getchar();
```

Δεν παίρνει κανένα όρισμα και επιστρέφει έναν ακέραιο.

# Χρήση της συνάρτησης getchar ( )

Για να διαβάσουμε έναν χαρακτήρα και να τον τυπώσουμε, γράφουμε:

```
#include <stdio.h>

int main() {

    printf("Gimme a char: ");

    int ch = getchar();

    if (ch != EOF) {

        printf("You gave the char: %c\n", ch);

    } else {

        printf("input ended\n");

    }

    return 0;

}
```

# Χρήση της συνάρτησης getchar()

Για να διαβάσουμε έναν χαρακτήρα και να τον τυπώσουμε, γράφουμε:

```
#include <stdio.h>

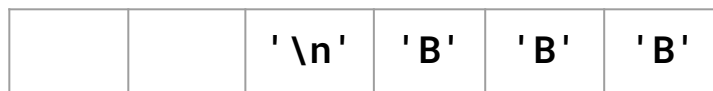
int main() {
    printf("Gimme a char: ");
    int ch = getchar();
    if (ch != EOF) {
        printf("You gave the char: %c\n", ch);
    } else {
        printf("input ended\n");
    }
    return 0;
}
```

Η getchar θα διαβάσει μόνο έναν χαρακτήρα

```
$ ./chartest
Gimme a char: BBB
You gave the char: B
```

# Χρήση της συνάρτησης `getchar()`

Συνήθως, πρέπει να περιμένουμε μέχρι να πατήσουμε **Enter** προκειμένου οι χαρακτήρες που πληκτρολογήσαμε να φτάσουν το πρόγραμμα.



Τα δεδομένα "αποθηκεύονται" προσωρινά σε έναν Buffer μέχρι να σταλεί καινούρια γραμμή και να προωθηθούν στο πρόγραμμα

# Χρήση της συνάρτησης `getchar()`

Για να δείξουμε ότι τελείωσαν τα δεδομένα εισόδου, στο Linux συνήθως πρέπει να πατήσουμε **Ctrl+D** (EOF)



Program

Όταν πατήσουμε **Ctrl+D** όλα τα δεδομένα που βρίσκονται στον `buffer` θα προωθηθούν στο πρόγραμμα (και χωρίς καινούρια γραμμή)



# Χρήση της συνάρτησης getchar ( )

Διαδοχικές κλήσεις της getchar() διαβάζουν διαδοχικούς χαρακτήρες. Τι κάνει το παρακάτω πρόγραμμα;

```
#include <stdio.h>

int main() {
    int ch, sum = 0;
    printf("Enter characters: ");
    while( (ch = getchar()) != '\n' && ch != EOF ) {
        printf("%c", ch);
        sum++;
    }
    printf("\nTotal characters: %d\n", sum);
    return 0;
}
```

# Χρήση της συνάρτησης getchar ( )

Διαδοχικές κλήσεις της getchar() διαβάζουν διαδοχικούς χαρακτήρες. Τι κάνει το παρακάτω πρόγραμμα;

```
#include <stdio.h>

int main() {
    int ch, sum = 0;
    printf("Enter characters: ");
    while( (ch = getchar()) != '\n' && ch != EOF ) {
        printf("%c", ch);
        sum++;
    }
    printf("\nTotal characters: %d\n", sum);
    return 0;
}
```

```
$ ./charcount
Enter characters: we'll always have paris
we'll always have paris
Total characters: 23
```

## Η συνάρτηση `putchar()`

Η συνάρτηση `putchar()` ορίζεται στο `stdio.h`, παίρνει έναν χαρακτήρα ως όρισμα, τον τυπώνει στο `stdout` του προγράμματος και τον επιστρέφει ως ακέραιο. Αν κάτι δεν πάει καλά στο τύπωμα, επιστρέφει την τιμή EOF (-1). Η συνάρτηση έχει την ακόλουθη μορφή:

```
int putchar(int c);
```

Προκειμένου να τυπώσουμε έναν χαρακτήρα 'C' απλά γράφουμε `putchar('C');`. Η συνάρτηση αυτή είναι ένα απλούστερο υποσύνολο της `printf`.

Τι πρόβλημα έχει η παρακάτω υλοποίηση της `cat`

```
#include <stdio.h>
```

```
int main() {  
    char c;  
    while((c = getchar()) != EOF)  
        putchar(c);  
    return 0;  
}
```

## Τι πρόβλημα έχει η παρακάτω υλοποίηση της `cat`

```
#include <stdio.h>
```

```
int main() {  
    char c;  
    while((c = getchar()) != EOF)  
        putchar(c);  
    return 0;  
}
```

```
$ echo -e "hello\xffworld" | ./cat  
hello  
$ echo -e "hello\xffworld" | cat  
hello world
```

Προσοχή: πάντα αναθέτουμε την τιμή επιστροφής της `getchar()` εκτός και αν είμαστε σίγουροι για το τι κάνουμε

## Τι κάνει το παρακάτω πρόγραμμα;

```
#define ERROR -1 // Return value for illegal character

int getinteger(int base) {

    char ch; // No need to declare ch as int - no EOF handling

    int val = 0; // Initialize return value

    while ((ch = getchar()) != '\n') // Read up to new line

        if (ch >= '0' && ch <= '0' + base - 1) // Legal character?

            val = base * val + (ch - '0'); // Update return value

        else

            return ERROR; // Illegal character read

    return val; // Everything OK - Return value of number read

}
```

# Τι κάνει το παρακάτω πρόγραμμα;

```
int i, ch, total = 0;

int letfr[26]; // Letter occurrences and frequencies array

for (i=0 ; i < 26 ; i++)
    letfr[i] = 0;

while ((ch = getchar()) != EOF) {
    if (ch >= 'A' && ch <= 'Z') {
        letfr[ch-'A']++;           // Found upper case letter
        total++;
    }
    if (ch >= 'a' && ch <= 'z') {
        letfr[ch-'a']++;           // Found lower case letter
        total++;
    }
}
```

# Για την επόμενη φορά

- Καλύψαμε έννοιες από τις σελίδες 28-29, 70-71, 86-87 από τις σημειώσεις του κ. Σταματόπουλου.
- [getchar](#) , [putchar](#)
- [Data buffer](#)
- [End of transmission](#)



Ευχαριστώ και καλή μέρα εύχομαι!  
Keep Coding ;)