Διάλεξη 9 - Δεδομένα Εισόδου

Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών

Εισαγωγή στον Προγραμματισμό

Θανάσης Αυγερινός / Τάκης Σταματόπουλος

Ανακοινώσεις / Διευκρινήσεις

- Χρήση τελεστή ισότητας με ορίσματα float / double
- Ονομασία συναρτήσεων camelCase vs snake_case

Τι θα τυπώσει το παρακάτω πρόγραμμα;

```c
#include <stdio.h>
int main() {
    float a = 3.1;

    if(a == 3.1)
       printf("OK\n");
    else
       printf("Not OK\n");
    return 0;
}
```

Υπάρχει κάποια εξήγηση;

Τι θα τυπώσει το παρακάτω πρόγραμμα;

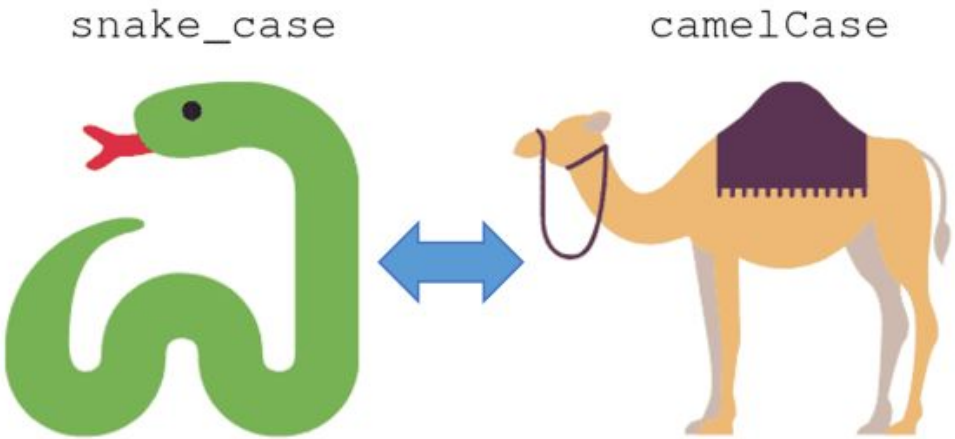
```c
#include <stdio.h>
int main() {
    float a = 3.1;

    if(a == 3.1)
       printf("OK\n");
    else
       printf("Not OK\n");
    return 0;
}
```

```
$ ./one
Not OK
```

Υπάρχει κάποια εξήγηση;

camelCase vs snake_case

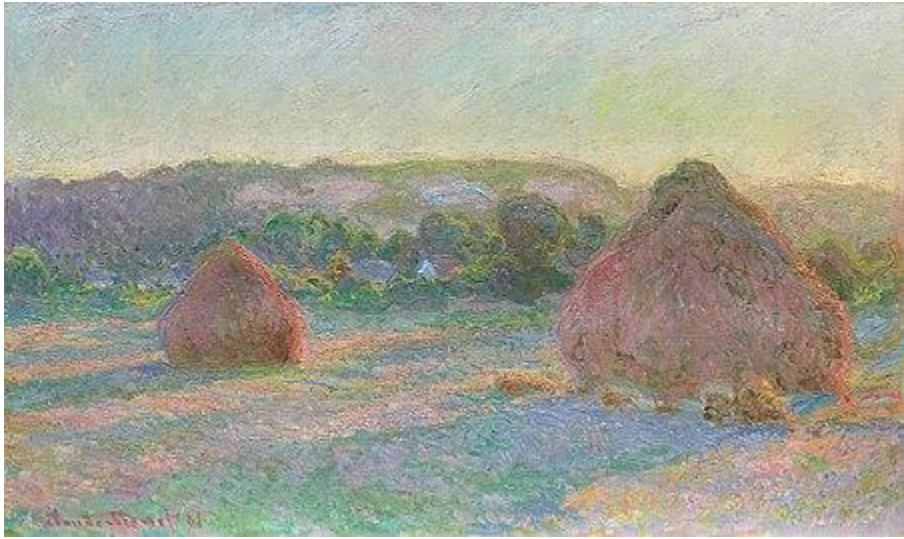


Για κώδικα C/Python συνιστώ `snake_case`

Παράδειγμα: `pi_approx` είναι καλύτερο (υποκειμενικό) από `piApprox`

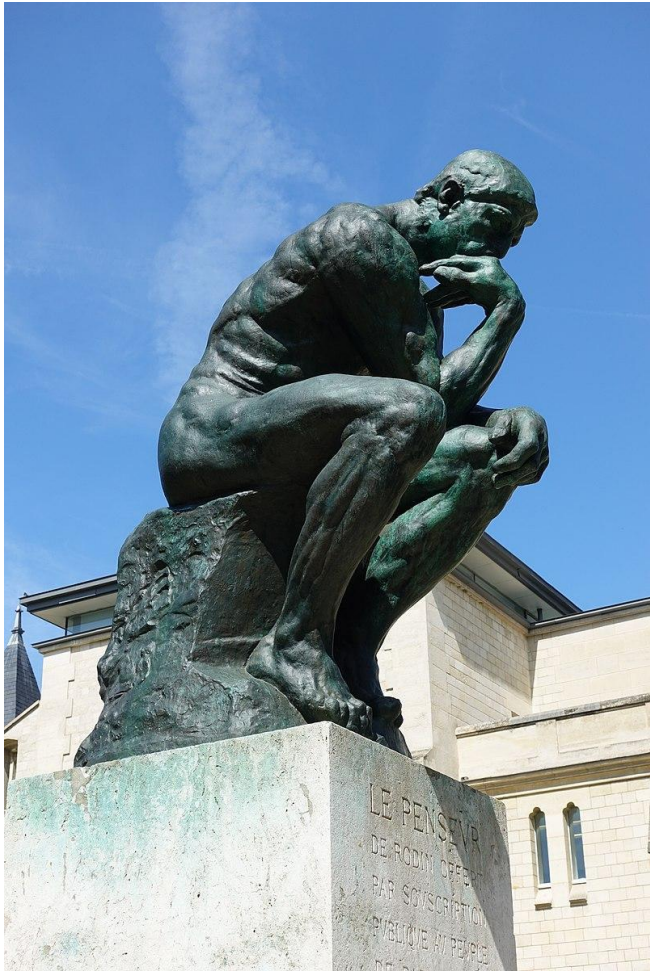
Την Προηγούμενη Φορά

- Παραδείγματα Επαναληπτικών Δομών
- Άλλες Δομές Ελέγχου
- Επίλυση Προβλημάτων



Σήμερα

- Δεδομένα Εισόδου και εξόδου
- Παραδείγματα

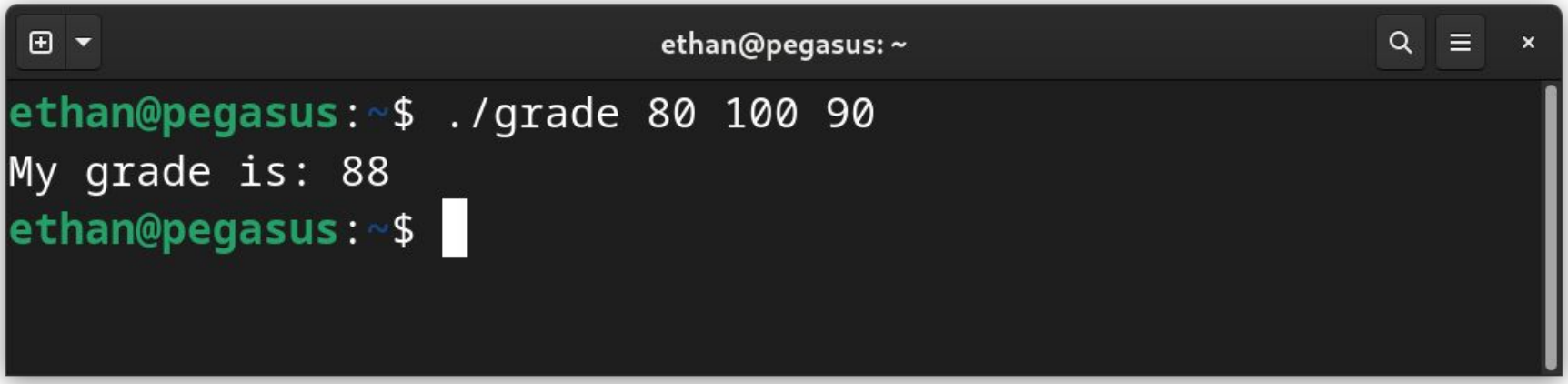


Δεδομένα Εισόδου και Εξόδου (Input and Output Data)

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:

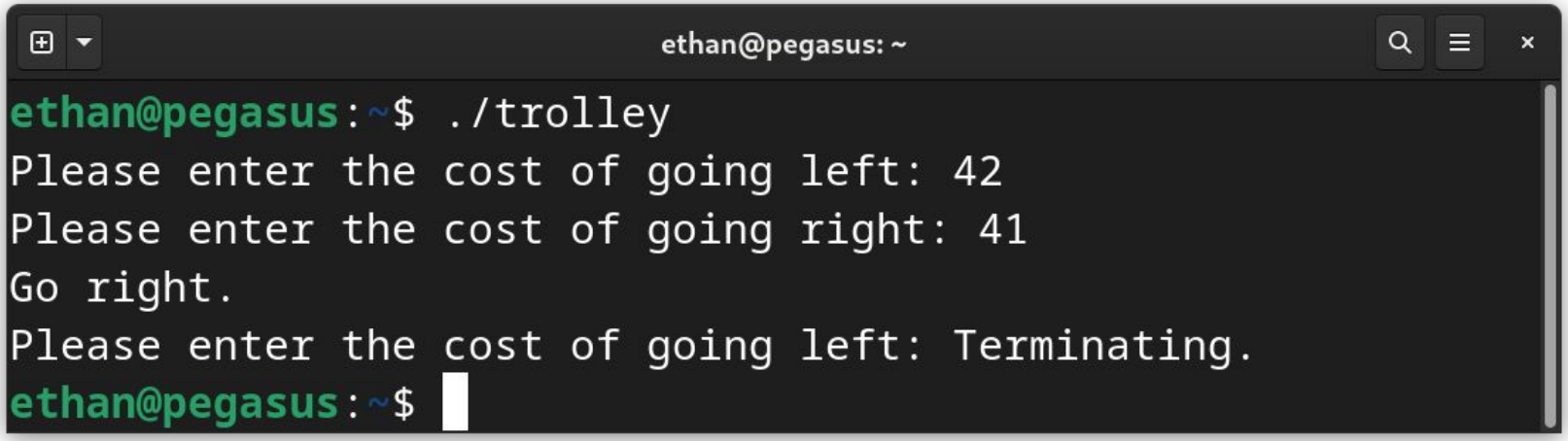
1. Ορίσματα στην γραμμή εντολών



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:

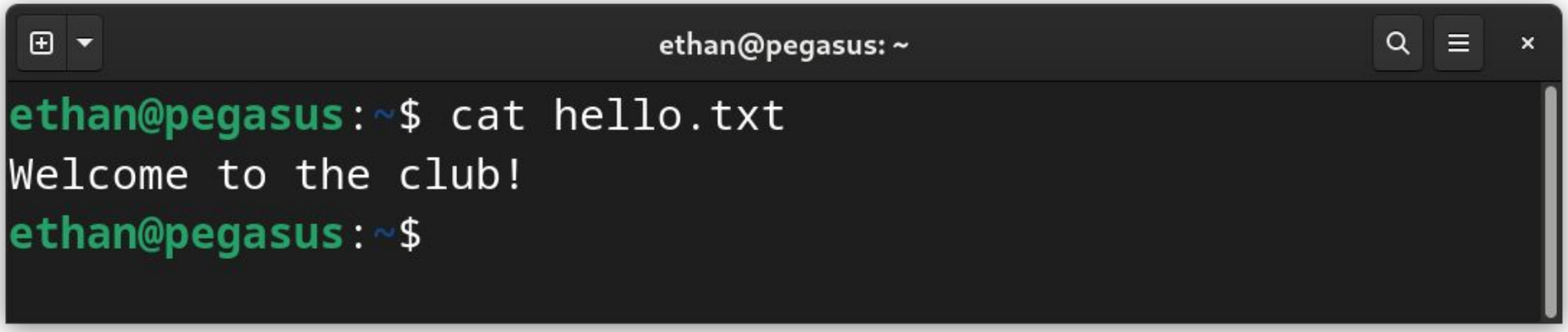
2. Γράφοντας κείμενο στην πρότυπη είσοδο (standard input ή stdin) συνήθως με το πληκτρολόγιο



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:

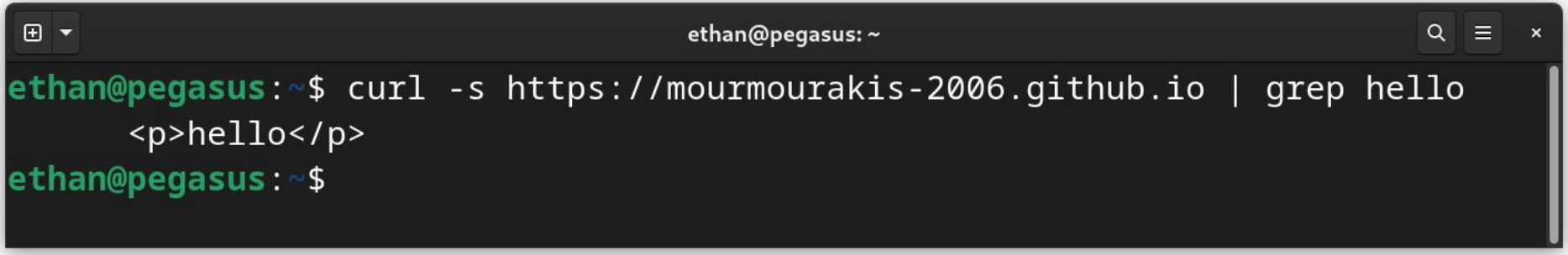
3. Διαβάζοντας αρχεία από το σύστημα αρχείων (επόμενες διαλέξεις)



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)



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)

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:

```c
#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:

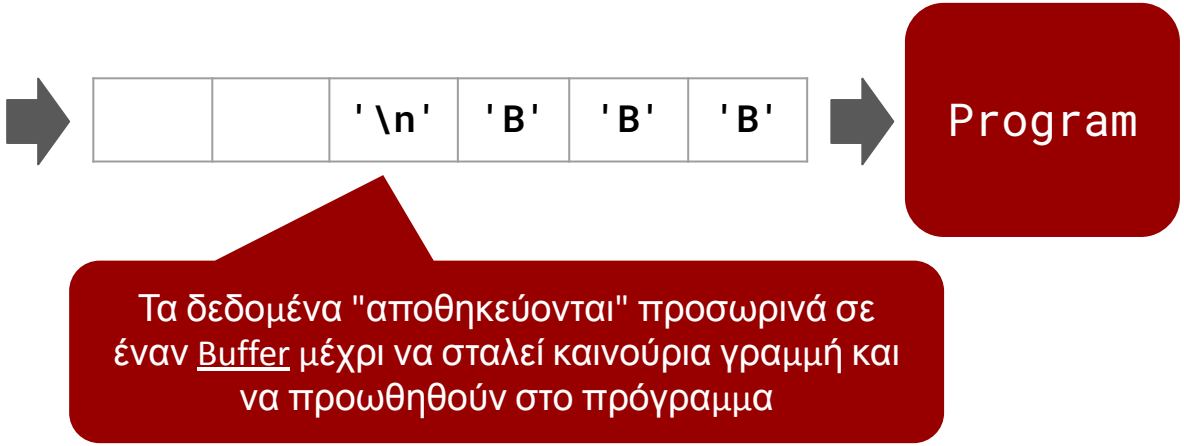
```c
#include <stdio.h>
int main() {
  printf("Gimme a char: ");
  int ch = getchar();
  if (ch != EOF) {
     printf("You gave the char: %c\n", ch);
  } else {
     printf("input ended\n");
  }
  return 0;
}
```

Η getchar θα διαβάσει μόνο έναν χαρακτήρα

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

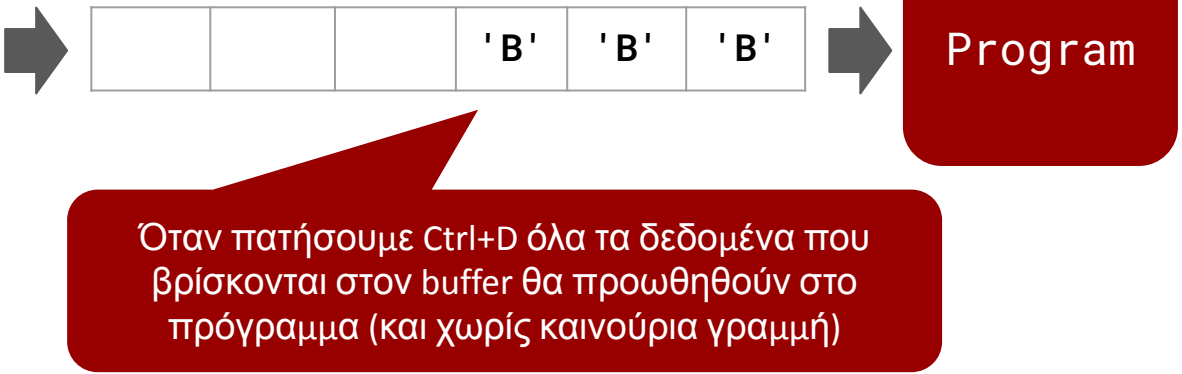
Χρήση της συνάρτησης `getchar()`

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



Χρήση της συνάρτησης `getchar()`

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



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()   ) != 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;

```c
#include <stdio.h>

int main() {

  int ch, sum = 0;

  printf("Enter characters: ");

  while( (ch = getchar()) != 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: 24
```

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 είναι ένα απλούστερο υποσύνολο της `printf`.

Τι πρόβλημα έχει η παρακάτω υλοποίηση της `cat`

```c
#include <stdio.h>

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

Τι πρόβλημα έχει η παρακάτω υλοποίηση της `cat`

```c
#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
helloworld
```

Προσοχή! Πρέπει να οριστεί ως **int**, όχι **char**

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

Θέλω να διαβάσω δύο αριθμούς από την πρότυπη είσοδο και να τους προσθέσω. Πως;

```
$ ./addnums
Give me a number: 40
Give me another number: 2
Total: 42
```

Θέλω να διαβάσω δύο αριθμούς από την πρότυπη είσοδο και να τους προσθέσω. Πως;

```
$ ./addnums
Give me a number: 40
Give me another number: 2
Total: 42
```

Κάνοντας χρήση της `getchar` και φτιάχνοντας μια συνάρτηση getinteger προκειμένου να διαβάσουμε τους χαρακτήρες έναν-έναν και να τους μετατρέψουμε σε αριθμό. Υπάρχει άλλος τρόπος να επιτύχουμε το ίδιο αποτέλεσμα;

```
#define ERROR -1
int getinteger(int base) {
  int ch;
  int val = 0;
  while ((ch = getchar()) != '\n')
    if (ch >= '0' && ch <= '0' + base - 1)
      val = base * val + (ch - '0');
    else
      return ERROR;
  return val;
}
```

Η συνάρτηση scanf

Η συνάρτηση scanf ορίζεται στο header file stdio.h και χρησιμοποιείται για να διαβάζει δεδομένα εισόδου πολλών τύπων από το stdin του προγράμματος και να αποθηκεύσει τις τιμές τους σε μεταβλητές. Αν επιτύχει, επιστρέφει πόσα δεδομένα εισόδου διάβασε. Αν αποτύχει, επιστρέφει την τιμή End-Of-File / EOF (-1).

Πως μπορώ να βρω πως συμπεριφέρεται;

Ανοίγω ένα τερματικό και τρέχω
`man scanf`!

Η συνάρτηση scanf

Η συνάρτηση scanf ορίζεται στο header file stdio.h και χρησιμοποιείται για να διαβάζει δεδομένα εισόδου πολλών τύπων από το stdin του προγράμματος και να αποθηκεύσει τις τιμές τους σε μεταβλητές. Αν επιτύχει, επιστρέφει *πόσα δεδομένα εισόδου* διάβασε. Αν αποτύχει, επιστρέφει την τιμή End-Of-File / EOF (-1). Η συνάρτηση έχει την ακόλουθη μορφή:

```
int scanf(const char *restrict format, ...);
```

Έχει μια συμβολοσειρά μορφοποίησης (format string)

Δέχεται όσα ορίσματα περάσουμε (άλλο μάθημα)

Η συνάρτηση scanf

Η συνάρτηση scanf ορίζεται στο header file stdio.h και χρησιμοποιείται για να διαβάζει δεδομένα εισόδου πολλών τύπων από το stdin του προγράμματος και να αποθηκεύσει τις τιμές τους σε μεταβλητές. Αν επιτύχει, επιστρέφει πόσα δεδομένα εισόδου διάβασε. Αν αποτύχει, επιστρέφει την τιμή End-Of-File / EOF (-1). Η συνάρτηση έχει την ακόλουθη μορφή:

```
int scanf(const char *restrict format, ...);
int printf(const char *restrict format, ...);
```

Είναι η συμμετρική της printf για διάβασμα αντί για εκτύπωση

Χρήση της συνάρτησης `scanf`

Τι κάνει το παρακάτω πρόγραμμα;

```c
#include <stdio.h>
int main() {
  int n;
  printf("Gimme a number: ");
  scanf("%d", &n);
  printf("Square: %d\n", n * n);
  return 0;
}
```

Χρήση της συνάρτησης scanf

Τι κάνει το παρακάτω πρόγραμμα;

```c
#include <stdio.h>
int main() {
  int n;
  printf("Gimme a number: ");
  scanf("%d", &n);
  printf("Square: %d\n", n * n);
  return 0;
}
```

Περνάμε την **διεύθυνση (spoiler)** της μεταβλητής n στην μνήμη ώστε η scanf να μπορέσει να αναθέσει την τιμή που διάβασε

Τυπώνει στο stdout το τετράγωνο του αριθμού που γράψαμε στο stdin

Χρήση της συνάρτησης scanf

Τι κάνει το παρακάτω πρόγραμμα;

```c
#include <stdio.h>
int main() {
  int n;
  printf("Gimme a number: ");
  scanf("%d", &n);
  printf("Square: %d\n", n * n);
  return 0;
}
```

Τι θα γινόταν αν γράφαμε `scanf("%d", n);` Γιατί;

```
$ ./scanf
Gimme a number: 3
Segmentation fault
```

Περνάμε την **διεύθυνση (spoiler)** της μεταβλητής n στην μνήμη ώστε η scanf να μπορέσει να αναθέσει την τιμή που διάβασε

Τυπώνει στο stdout το τετράγωνο του αριθμού που γράψαμε στο stdin

Χρήση της συνάρτησης `scanf`

Τι κάνει το παρακάτω πρόγραμμα;

Είναι σωστό αυτό το πρόγραμμα;

```c
#include <stdio.h>
int main() {
  int n;
  printf("Gimme a number: ");
  scanf("%d", &n);
  printf("Square: %d\n", n * n);
  return 0;
}
```

Χρήση της συνάρτησης scanf

Τι κάνει το παρακάτω πρόγραμμα;

```c
#include <stdio.h>
int main() {
  int n;
  printf("Gimme a number: ");
  scanf("%d", &n);
  printf("Square: %d\n", n * n);
  return 0;
}
```

Είναι σωστό αυτό το πρόγραμμα;

Όχι καθώς δεν ελέγχουμε την τιμή επιστροφής της scanf (EOF ή ίσως 0!)

Χρήση της συνάρτησης scanf

Τι κάνει το παρακάτω πρόγραμμα;

```c
#include <stdio.h>
int main() {
  int n;
  printf("Gimme a number: ");
  scanf("%d", &n);
  printf("Square: %d\n", n * n);
  return 0;
}
```

```
$ ./scanf
Gimme a number: 16
Square: 256
$ ./scanf
Gimme a number: Square:
1068701481
$ ./scanf
Gimme a number: hello
Square: 1072038564
```

Χρήση της συνάρτησης `scanf` - Πολλά ορίσματα

Τι κάνει το παρακάτω πρόγραμμα;

```c
#include <stdio.h>
int main() {
  int n1, n2;
  printf("Gimme two numbers: ");
  scanf("%d %d", &n1, &n2);
  printf("Result: %d\n", n1 * n2);
  return 0;
}
```

Χρήση της συνάρτησης `scanf` - Πολλά ορίσματα

Τι κάνει το παρακάτω πρόγραμμα;

```c
#include <stdio.h>
int main() {
  int n1, n2;
  printf("Gimme two numbers: ");
  scanf("%d %d", &n1, &n2);
  printf("Result: %d\n", n1 * n2);
  return 0;
}
```

```
$ ./scanf2
Gimme two numbers:   2   4
Result: 8
```

Καθώς ψάχνει για δεκαδικό ψηφίο, η scanf αγνοεί τους κενούς χαρακτήρες ή αλλαγές γραμμής

Τι κάνει το παρακάτω πρόγραμμα;

```c
#define ERROR -1                              // Return value for illegal character
int getinteger(int base) {
  int 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, 78-79, 86-87 από τις σημειώσεις του κ. Σταματόπουλου.
- getchar , putchar, scanf, printf
- Data buffer
- End of transmission
- Canonical mode

Ευχαριστώ και καλή μέρα εύχομαι!
Keep Coding ;)