

printf

- Οι κωδικοί (format codes) του printf για διάφορους τύπους δεδομένων είναι:
 - `%d` για ακεραίους (int)
 - `%lf` για κινητής υποδιαστολής διπλής ακρίβειας (double)
 - `%f` για κινητής υποδιαστολής απλής ακρίβειας (float)
 - `%c` για χαρακτήρες (char)
 - `%s` για συμβολοσειρές

printf

```
printf ( "Value = %d\n" , num );
```

Τι θέλουμε να εμφανιστεί στην οθόνη:

η λέξη Value

ένα κενό

ένα =

ένα κενό

ένας **ακέραιος**

ένας χαρακτήρας αλλαγής γραμμής

Η τιμή που θα εμφανιστεί

στη θέση του ακεραίου

που προσδιορίσαμε

με το %d.

Εδώ μπορεί να τοποθετηθεί

οποιαδήποτε έκφραση

αρκεί το αποτέλεσμα

να έχει ακέραια τιμή

printf

- Με τη χρήση ειδικών κωδικών (format modifiers) μπορούμε να προσδιορίσουμε ακριβώς πώς θέλουμε να εμφανιστεί ένας αριθμός
- `%6d` : Το πλάτος του ακεραίου είναι 6 θέσεις.
 - Αν ο ακέραιος έχει λιγότερα από έξι ψηφία, τότε οι επιπλέον θέσεις γεμίζουν με κενά.
 - Αν θέλουμε οι επιπλέον θέσεις να γεμίσουν με μηδενικά, τότε γράφουμε `%06d`

printf παράδειγμα

```
#include<stdio.h>

int main (int argc, char *argv[]) {
    int num = 46;
    printf ("num=%5d\n", num);
    printf ("num=%05d\n", num);
    return 0;
}
```

Έξοδος:

```
>num=    46
>num=00046
>
```

printf

- %6.2f : Το πλάτος του αριθμού κινητής υποδιαστολής είναι 6 θέσεις και τα δεκαδικά ψηφία είναι 2.
- Προσοχή: 6 είναι το πλάτος ΟΛΟΥ του αριθμού (ακέραιο μέρος + μια θέση για την υποδιαστολή + δεκαδικό μέρος)

printf παράδειγμα

```
#include<stdio.h>

int main (int argc, char *argv[]) {
    double num = 1.234567;
    double longnum = 1234567.5;
    printf ("num=%6.2lf\n", num);
    printf ("num=%6.2lf\n", longnum);
    return 0;
}
```

Έξοδος:

```
>num=  1.23
>num=1234567.50
>
```

Το πλάτος που προσδιορίσαμε δεν ήταν αρκετό. Το ακέραιο μέρος εκτυπώθηκε σε όσες θέσεις χρειαζόταν, όχι όσες ζητήσαμε.

printf παράδειγμα

Θέλουμε να εμφανιστεί στην οθόνη το μήνυμα:
The tax rate is 9.5%.
όπου το 9.5 είναι αποθηκευμένο σε μια σταθερά.

```
#include<stdio.h>

int main (int argc, char *argv[]) {
    const double taxrate = 9.5;
    printf ("The tax rate is %.11f%%.\n", taxrate);

    return 0;
}
```

Ο χαρακτήρας % έχει συγκεκριμένο νόημα για το printf: χρησιμοποιείται μαζί με έναν κωδικό (πχ. d, lf) για να δηλώσει ότι σε αυτό το σημείο πρέπει να εμφανιστεί η τιμή μιας έκφρασης. Αν θέλουμε απλά να εκτυπώσουμε το "επι τοις εκατό" τότε χρησιμοποιούμε δύο %

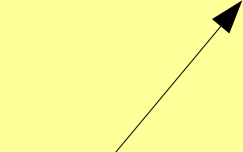
printf παράδειγμα

Θέλουμε να εμφανιστεί στην οθόνη το μήνυμα:
My name is "John".

```
#include<stdio.h>

int main (int argc, char *argv[]) {
    printf ("My name is \"John\".\n");

    return 0;
}
```



Ο χαρακτήρας " έχει συγκεκριμένο νόημα για το printf: περικλείει το κείμενο που εκτυπώνεται στην οθόνη.
Αν θέλουμε απλά να εκτυπώσουμε ένα " τότε χρησιμοποιούμε ένα \ πριν το "

scanf

- Οι κωδικοί (format specifiers) του scanf για διάφορους τύπους δεδομένων είναι:
 - `%d` για ακεραίους (int)
 - `%lf` για κινητής υποδιαστολής διπλής ακρίβειας (double)
 - `%f` για κινητής υποδιαστολής απλής ακρίβειας (float)
 - `%c` για χαρακτήρες (char)
 - **Προσοχή:** Σε αυτή την περίπτωση πρέπει πάντα να παρεμβάλεται ένα κενό μεταξύ " και %

scanf παραδείγματα

```
#include<stdio.h>

int main (int argc, char *argv[]) {
    double base;
    scanf("%lf", &base);
    printf("%lf\n", base);
    return 0;
}
```

Είσοδος

2.17

Έξοδος

2.17

scanf παραδείγματα

```
#include<stdio.h>

int main (int argc, char *argv[]) {
    double num;
    scanf("num = %lf", &num);
    printf("%lf\n", num);
    return 0;
}
```

Είσοδος

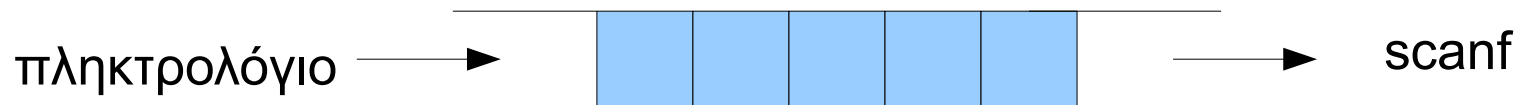
num = 2.17

Έξοδος

2.17

scanf

- Πώς λειτουργεί το scanf?
- Ό,τι γράφετε στο πληκτρολόγιο αποθηκεύεται πρώτα σε μια περιοχή προσωρινής αποθήκευσης (buffer) και από εκεί διαβάζει το scanf.



scanf

- Αν ο buffer είναι άδειος, τότε το scanf περιμένει μέχρι να μπουν χαρακτήρες σε αυτόν.
 - Όταν ο χρήστης γράψει κάτι στο πληκτρολόγιο, τότε αυτό αποθηκεύεται στο buffer και το scanf αρχίζει να διαβάζει.
- Αν ο buffer περιέχει κάτι αλλά είναι λάθος τύπος, τότε το scanf αποτυγχάνει στην ανάγνωση και "επιστρέφει" χωρίς να έχει διαβάσει κάτι.
 - Αν το scanf συνεχίσει να προσπαθεί να διαβάσει το ίδιο πράγμα (πχ. σε μια επανάληψη), τότε το πρόγραμμα θα "κολήσει"

scanf παραδείγματα

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

```
int main (int argc, char *argv[]) {  
    int num;  
    scanf("%d", &num);  
    printf("num = %d\n", num);  
    return 0;  
}
```

Είσοδος:

d

Έξοδος:

num = -1208857696

Το scanf περίμενε ακέραιο και του δώσαμε χαρακτήρα. Δε διαβάστηκε τίποτα μέσα στο num, οπότε εκτυπώθηκαν σκουπίδια.

scanf παραδείγματα

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

```
int main (int argc, char *argv[]) {  
    int num;  
    scanf("%d", &num);  
    printf("num = %d\n", num);  
    return 0;  
}
```

Είσοδος:

34.7

Έξοδος:

num = 34

Το scanf περίμενε ακέραιο και
του δώσαμε δεκαδικό.
Το scanf διάβασε μόνο το ακέραιο
κομμάτι και το .7 έμεινε
στο buffer.

scanf παραδείγματα

```
#include<stdio.h>

int main (int argc, char *argv[]) {
    int aem;
    char grade;
    float score;
    scanf("%d%c%f", &aem, &grade, &score);
    printf("%d: %c, %.1f\n", aem, grade, score);
    return 0;
}
```

Είσοδος:

123C6.5

Έξοδος:

123: C, 6.5

scanf παραδείγματα

```
#include<stdio.h>

int main (int argc, char *argv[]) {
    int aem;
    char grade;
    float score;
    scanf("%d %c %f", &aem, &grade, &score);
    printf("%d: %c, %.1f\n", aem, grade, score);
    return 0;
}
```

Είσοδος:

123CD6.5

Έξοδος:

123: C, 0.0

scanf παραδείγματα

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

```
int main (int argc, char *argv[]) {  
    int num;  
    char grade;  
    scanf ("%d", &num);  
    scanf ("%c",&grade);  
    printf ("**%c**\n", grade);  
    return 0;  
}
```

Είσοδος:

>15
>A

Έξοδος:

**
**

Τι συνέβη? Αντί να εκτυπώσει A όπως περιμέναμε, τύπωσε ένα χαρακτήρα αλλαγής γραμμής!
'Όταν εισάγαμε το 15, μπήκαν δύο πράγματα στο buffer: το 15 και το Enter που πατήσαμε. Το πρώτο scanf διάβασε το 15, και το enter έμεινε στο buffer. Το δεύτερο scanf διάβασε το enter.

scanf παραδείγματα

Για να αποφύγουμε το πρόβλημα που περιγράφεται στην προηγούμενη διαφάνεια, γράφουμε

```
scanf(" %c", &grade);
```

(δηλαδή βάζουμε ένα κενό ανάμεσα στο " και στο %)

Αυτό λέει στο scanf ότι θέλουμε να διαβάσει "white space" και μετά ένα χαρακτήρα.

Θα διαβάσει το enter (το οποίο είναι white space) και μετά το Α κανονικά.

scanf παραδείγματα

- Γιατί δεν παρουσιάζεται το ίδιο πρόβλημα όταν διαβάζουμε αριθμούς?
- Διότι όταν το scanf πρόκειται να διαβάσει ένα αριθμό, αγνοεί το white space που μπορεί να περιέχει ο buffer πριν από τον αριθμό.
 - Αλλά δεν αγνοεί άλλους χαρακτήρες
 - white space είναι τα κενά, enter, tab.