ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ
UNIVERSITY OF PATRAS

ΑΝΟΙΚΤΑ ακαδημαϊκά μαθήματα ΠΠ

ΑΡΧΕΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

Κεφάλαιο 2

Επιμέλεια:
Βασίλης Παλιουράς , Αναπληρωτής Καθηγητής
Ευάγγελος Δερματάς , Αναπληρωτής Καθηγητής
Σταύρος Νούσιας , Βοηθός Ερευνητή
Πολυτεχνική Σχολή
Τμήμα Ηλεκτρολόγων Μηχανικών & Τεχνολογίας Υπολογιστών

Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.
- Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τυπου άδειας χρησης, η άδεια χρησης αναφερεται ρητώς.

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο
ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ
ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ
Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου των διδασκόντων καθηγητών.
- Το έργο «Ανοικτά Ακαδήμαϊκά Μαθήματα στο Πανεπιστήμιο Πατρών » έχει χρηματοδοτηθεί μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκη Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικους πόρους.

Ανάπτυξη

Το παρόν εκπαιδευτικό υλικό αναπτύχθηκε στο τμήμα Ηλεκτρολόγων Μηχανικών και Τεχνολογίας Υπολογιστών του Πανεπιστημίου Πατρών

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο
ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ
ΕΣΠΑ 2007-2013
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ
Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Μερικές διαφορές python, C

- Στατικοί τύποι – δυναμικοί τύποι
  - Διαθέσιμοι ή κατασκευασμένοι
- Οργάνωση προγράμματος
  - Κώδικας μόνο σε συναρτήσεις
    - Επιτρέπονται δηλώσεις εκτός συναρτήσεων
  - Υποχρεωτικά δηλώσεις **ονομάτων**
    - Τύπων, μεταβλητών, …
    - Κατάλληλες δηλώσεις αξιοποιούν το **type checking**
  - Μπλοκ κώδικα με { }

    αντί στοίχιση ανά γραμμή
- Θα δούμε και άλλες…

Αξιοποίηση αφαιρετικότητας με διεργασίες

```c
#include <stdio.h>
int computeGCD(int, int);

int main ( ) {
 int a, b;
 int gcd;

 scanf ("%d %d", &a, &b) ;

 gcd = computeGCD (a, b);

 printf("%d\n", gcd);
}
```

```c
int computeGCD(int a , int b)
{
int result;
int i = a, j = b;

 while (i != j) {
   if ( i > j)
        i = i - j ;
     else
        j = j - i;
   }

return i;
}
```

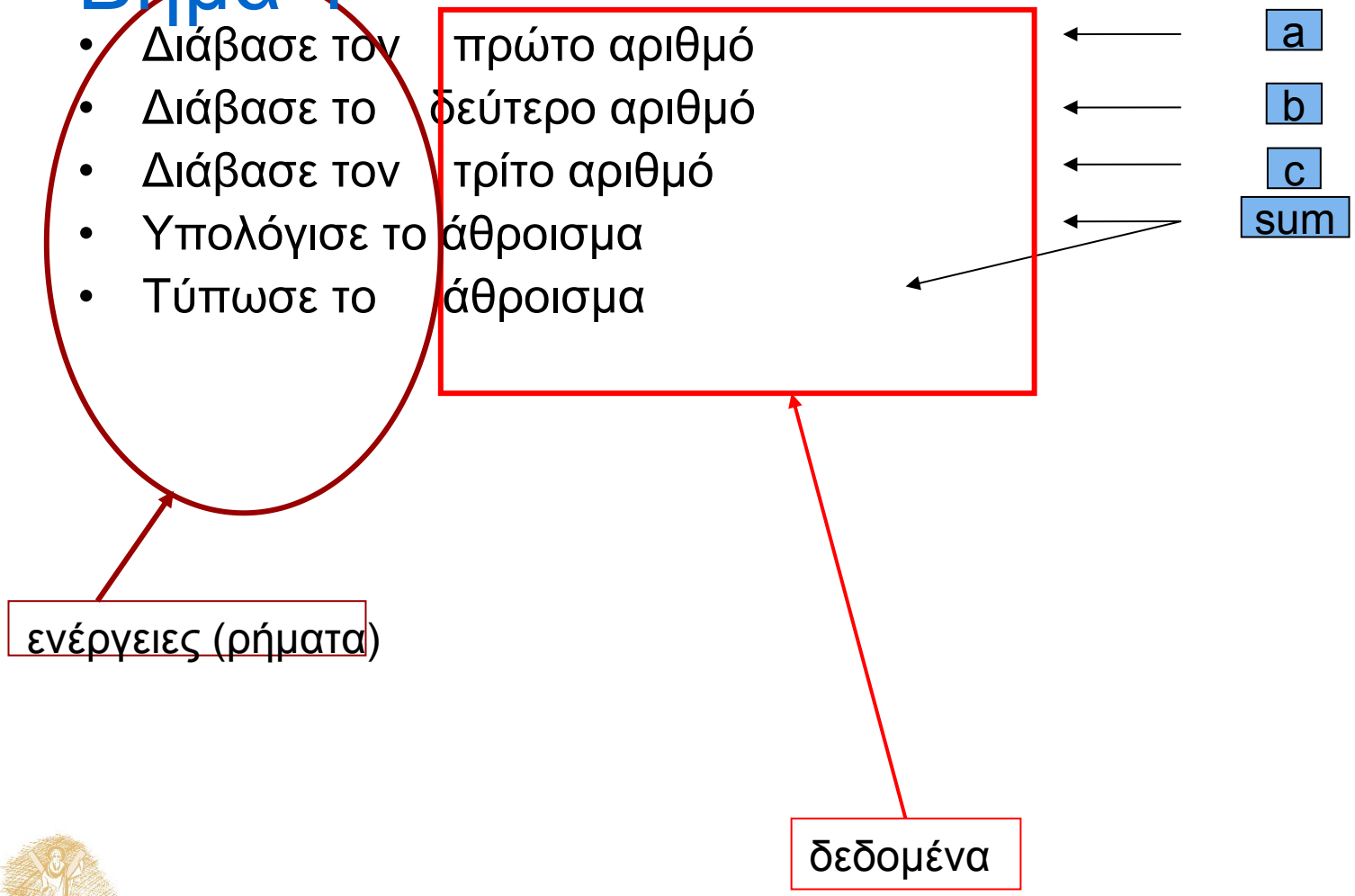
Dev-cpp

- File > New > Project
- Console application
- C project
- Command line
- gcc test.c –o test.exe

Πρώτο Πρόγραμμα σε C

Βήμα 1

- Διάβασε τον πρώτο αριθμό
- Διάβασε το δεύτερο αριθμό
- Διάβασε τον τρίτο αριθμό
- Υπολόγισε το άθροισμα
- Τύπωσε το άθροισμα



Πρώτο Πρόγραμμα σε C

Ρήματα => διεργασίες => κλήσεις συναρτήσεων

```
int a, b, c, sum;
```

- Διάβασε τον a ← `scanf("%d", &a);`
- Διάβασε το b ← `scanf("%d", &b);`
- Διάβασε τον c ← `scanf("%d", &c);`
- Υπολόγισε το sum ← `sum = a + b + c;`
- Τύπωσε το sum ← `printf("the sum is: %d\n", sum);`

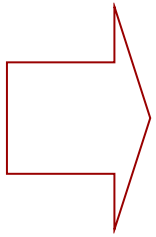
Δεύτερο Πρόγραμμα σε C

```
#include <stdio.h>

main ( ) {
   int a, b, c, sum;

   scanf("%d", &a);
   scanf("%d", &b);
   scanf("%d", &c);

   sum = a + b + c;

   printf ("sum is %d", sum);
}
```

τι κάνει;

Καλές επιλογές ονομάτων => Ευανάγνωστος κώδικας

```
i= 120;
if (i>j)
func1();
else
func2();
```



```
velocity = 120;
if (velocity > max_velocity)
   decrease_velocity( );
else
   increase_velocity( );
```

Μέχρι τώρα...

- Αφαιρετικότητα
  - Ως εργαλείο για την αντιμετώπιση της πολυπλοκότητας
- Σχεδιασμός Προγράμματος
- Αναφέραμε την Αυξητική Ανάπτυξη Προγράμματος
  - Υποστηρίζει top-down ανάπτυξη
  - Διευκολύνει την ανάπτυξη του προγράμματος
  - διευκολύνει τον έλεγχο του κώδικα

Βασική μέθοδος

- Αυξητική ανάπτυξη προγράμματος (incremental development)
- Να γραφεί ένα πρόγραμμα που διαβάζει έναν αριθμό, να υπολογίζει την τρίτη δύναμή του, και στη συνέχεια να τυπώνει το αποτέλεσμα.

Παράδειγμα

- Διάβασε έναν αριθμό ← number
- Υπολόγισε την τρίτη δύναμή του ← power
- Τύπωσε το αποτέλεσμα ← power

- **int** number, power;
- Διάβασε number ← diabase()
- Υπολόγισε power ← ypologisepower()
- Τύπωσε power ← typwse()

Σχεδίαση top-down

```
main( ) {
  int number, power;
  number = diabase( );
  power = ypologisepower(number );
  typwse (power );

}
```

Έκδοση 0: θα πρέπει να είναι εκτελέσιμο!

```c
#include <stdio.h>
int diabase();
int ypologisepower(int);
void typwse(int);

main( ) {
 int number, power;
 number = diabase( );
 power = ypologisepower(number);
 typwse(power);

}
```

```c
int diabase ( ) {
 printf ("function: diabase\n");
 return 5;
}

int ypologisepower(int x ) {
 printf ("function: ypologise\n");
 return x;
}

void typwse (int x ) {
  printf("function: typwse\n");
}
```

Έκδοση 2: πλήρης ypologisepower()

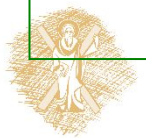
```c
#include <stdio.h>
int diabase();
int ypologisepower(int);
void typwse(int);

main( ) {
 int number, power;
 number = diabase( );
 power = ypologisepower(number);
 typwse(power);

}
```

```c
int diabase ( ) {
 printf ("function: diabase\n");
 return 5;
}

int ypologisepower(int x ) {
 printf ("function: ypologise\n");
 return x * x * x;
}

void typwse (int x ) {
 printf("function: typwse\n");
 printf("apotelesma: %d\n",x);
}
```

Έκδοση 3: πλήρης diabase()

```c
#include <stdio.h>
int diabase();
int ypologisepower(int);
void typwse(int);

main( ) {
 int number, power;
 number = diabase( );
 power = ypologisepower(number);
 typwse(power);

}
```

```c
int diabase ( ) {
 int aninput;
 printf ("function: diabase\n");
 scanf("%d", &aninput);
 return aninput;
}

int ypologisepower(int x ) {
 printf ("function: ypologise\n");
 return x * x * x;
}

void typwse (int x ) {
 printf("function: typwse\n");
 printf("apotelesma: %d\n",x);
}
```



Στοιχεία της Γλώσσας C

- Γραμματική και Συντακτικό
- Διαθέσιμοι τύποι δεδομένων
  - Απλοί και σύνθετοι τύποι
- Βασική βιβλιοθήκη της C
  - παρέχει ένα σύνολο έτοιμων συναρτήσεων: printf( ), scanf( ), …
- Εκτεταμένη τεκμηρίωση της GNU C library
  - http://www.gnu.org/software/libc/manual/

Δεσμευμένες λέξεις (reserved words)

- Λέξεις κλειδιά (keywords)
- Ονόματα συναρτήσεων της βασικής βιβλιοθήκης
- Ονόματα μακροεντολών που ορίζονται σε αρχεία επικεφαλίδας: EOF, …
- Ονόματα τύπων που ορίζει η βασική βιβλιοθήκη: time_t, …
- Ονόματα εντολών προεπεξεργαστή: include, define
- Ονόματα της μορφής _DATE_, _FILE_, κτλ.

Αναγνωριστές (Identifiers)

- Λέξεις που κατασκευάζει ο προγραμματιστής για να ονομάσει
  - μεταβλητές
  - σταθερές
  - συναρτήσεις
  - ...
- Δεν θα πρέπει να είναι δεσμευμένες

Τύποι Δεδομένων στη C

- char – χαρακτήρας
- int – ακέραιος
- float – αριθμός κινητής υποδιαστολής απλής ακρίβειας
- double – αριθμός κινητής υποδιαστολής διπλής ακρίβειας
- απαριθμητικός τύπος
  - enum boolean {FALSE, TRUE};
- σύνθετοι τύποι
  - πίνακες και δομές (struct)

Παραδείγματα Αναγνωριστών - ποιοί είναι σωστοί;

- j
- 5j
- lname
- _Fname
- $amount
- set_password
- int
- MaXveLocity
- find_max#of_lements
- get_word
- isdigit
- get@name

δοκιμή

Μετατροπή Fahrenheit σε Celcius

```
#include <stdio.h>
/* print Fahrenheit-Celcius table for
   fahr = 0, 20, …, 300
*/

main ( ) {
      int fahr, celcius;
      int lower, upper, step;

      lower = 0 ; /* lower limit of temperature */
      upper = 300;
      step = 20 ;
      fahr = lower ;
      while (fahr <= upper) {
      celcius = 5* (fahr – 32) / 9;
      printf ("%d\t%d\n", fahr, celcius);
      fahr = fahr + step;
}
}
```

Επεξήγηση

```
#include <stdio.h>
/* print Fahrenheit-Celcius table for fahr = 0, 20, …, 300 */
main () {
int fahr, celcius;
int lower, upper, step;
lower = 0 ; /* lower limit of temperature */
upper = 300;
step = 20 ;
fahr = lower ;
while (fahr <= upper) {
celcius = 5* (fahr – 32) / 9;
printf (“%d\t%d\n”, fahr, celcius);
fahr = fahr + step;
}
}
```

δήλωση μεταβλητών

Επεξήγηση

```
#include <stdio.h>
/* print Fahrenheit-Celcius table for fahr = 0, 20, …, 300 */

main () {

int lower = 0, upper = 300, step = 20;
int fahr = lower, celcius;

while (fahr <= upper) {
celcius = 5* (fahr - 32) / 9;
printf ("%d\t%d\n", fahr, celcius);
fahr = fahr + step;
}

}
```

Δήλωση μεταβλητών

Μαζί με αρχικοποίηση

Επεξήγηση

```
#include <stdio.h>
/* print Fahrenheit-Celcius table for fahr = 0, 20, …, 300 */

main () {

int lower = 0, upper = 300, step = 20;
int fahr = lower, celcius;

while (fahr <= upper) {
celcius = 5* (fahr - 32) / 9;
printf ("%d\t%d\n", fahr, celcius);
fahr = fahr + step;
}

}
```

Χρήση άλλης μεταβλητής

Για αρχικοποίηση

Σημείωμα αναφοράς

- Copyright Πανεπιστήμιο Πατρών, Παλιουράς Βασίλειος , Δερματάς Ευάγγελος «Αρχές Προγραμματισμού ». Έκδοση: 1.0. Πάτρα 2015
- Διαθέσιμο από τη δικτυακιακή διέυθυνση https://eclass.upatras.gr/modules/