Δομημένος Προγραμματισμός

Τμήμα Επιχειρηματικού Σχεδιασμού και Πληροφοριακών Συστημάτων

www.bpis.teicrete.gr

Νέο Πρόγραμμα

- Δημιουργήστε ένα νέο project με τίτλο

Operators2

Άσκηση

- **Γράψτε ένα πρόγραμμα που τυπώνει τη θερμοκρασία σε βαθμούς Fahreneit που αντιστοιχεί στη θερμοκρασία 21 C. Ο τύπος που θα χρησιμοποιήσετε είναι Βαθμοί F = 9/5*βαθμοί C + 32**
  - Προσέξτε τον τύπο των μεταβλητών που θα χρησιμοποιήσετε (int? float?)
  - Δώστε ξεκάθαρα ονόματα στις μεταβλητές
  - Στοιχίστε σωστά τον κώδικά σας
  - Βάλτε σχόλια

Άσκηση

- **Αλλάξτε το προηγούμενο πρόγραμμα που γράψατε ούτως ώστε να διαβάζει μια θερμοκρασία από το πληκτρολόγιο και μετά να τυπώνει την αντίστοιχή της σε Fahreneit**

Η συνάρτηση scanf

- Η εισαγωγή τιμών στις μεταβλητές μπορεί να γίνει με τη χρήση της συνάρτησης scanf και του αντίστοιχου προσδιοριστή (specifier)

π.χ. scanf(" %c" , &ch);

- Επιστρέφει:
  - τον αριθμό των στοιχείων που διάβασε επιτυχώς.
  - 0 αν δεν διαβάσει τίποτα
  - μια ειδική τιμή, την EOF (συνήθως ισούται με –1), αν βρεθεί σε κατάσταση end of file

- **Προσδιοριστές:**
  - %c χαρακτήρας %d,%i ακέραιος
  - %f κινητής υποδιαστολής %lf διπλής ακρίβειας
  - %x δεκαεξαδικός %s συμβολοσειρά

Τελεστές αύξησης και μείωσης

- Τελεστής αύξησης (increment operator) συμβολίζεται με **++**
  π.χ num = num + 1;
  ισοδυναμεί με num++;

- Τελεστής μείωσης (decrement operator) συμβολίζεται με **--**
  πχ. num = num – 1;
  Ισοδυναμεί με num--;

- Οι τελεστές αύξησης και μείωσης μπορούν να δυσκολέψουν την κατανόηση πράξεων κυρίως όταν παρουσιάζονται ως προπορευόμενοι ή παρελκόμενοι

π.χ. y = x-- + y;
z = ++x + y;

Τελεστές αύξησης και μείωσης & προτεραιότητα

- Σε περίπτωση προπορευόμενου τελεστή το σύστημα πρώτα εκτελεί την αύξηση ή μείωση και μετά χρησιμοποιεί την νέα τιμή της μεταβλητής στην έκφραση

πχ. z=++x + y;

Τελεστές αύξησης και μείωσης & προτεραιότητα

- Αντιθέτως, στην περίπτωση του παρελκόμενου τελεστή το σύστημα πρώτα χρησιμοποιεί την τιμή της μεταβλητής για τον υπολογισμό της τιμής της έκφρασης και μετά εκτελεί την αύξηση ή μείωση της τιμής της μεταβλητής

πχ. z= x++ +y;

Νέο Πρόγραμμα

- Δημιουργήστε ένα νέο project με τίτλο

Operators3

Άσκηση

```
 int x, y,z;
x = 10;
y = 20;

 z = ++x + y;
printf(" z = %d \n",z);

x = 10;
y = 20;

 z = x++ + y;
printf(" z = %d \n",z);
```

Τελεστές ανάθεσης

- Σκοπός αυτών είναι να εκτελούν κάποια πράξη ανάμεσα στους τελεστέους και να εκχωρούν το αποτέλεσμα σε έναν από τους τελεστέους

  +=, -=, /=, %=, *=

  πχ. x *= 10 ισοδυναμεί με x = x * 10;

- Στην κατηγορία των τελεστών ανάθεσης έχουμε και τους τελεστές διαχείρισης δυαδικών ψηφίων (bitwise operator)

  >>= <<= &= ^= |=

Νέο Πρόγραμμα

- Δημιουργήστε ένα νέο project με τίτλο

Operators4

Άσκηση

Γράψτε το παρακάτω πρόγραμμα και αντικαταστήστε όπου είναι δυνατό με σύνθετους τελεστές ανάθεσης

```
int x=10, y=20, z=0;


z = z+x;
x = x*10;
printf(" z = %d \n",z);
z=x+y;
z=z%x;
y=y-x;
printf(" z = %d \n",z);
printf(" y = %d \n",y);
```

Συσχετιστικοί Τελεστές (relational operators)

- Συγκρίνουν δύο τελεστέους

- Το αποτέλεσμα της έκφρασης με συσχετιστικούς τελεστές είναι **ΑΛΗΘΕΣ (1)** ή **ΨΕΥΔΕΣ (0)**

  ➔ π.χ 3 < 2 επιστρέφει 0

  ➔ π.χ. 2 == 2 επιστρέφει 1

Συσχετιστικοί Τελεστές (relational operators)

Συσχετιστικοί Τελεστές	Δράση
<	Μικρότερο
>	Μεγαλύτερο από
<=	Μικρότερο ή ίσον από
>=	Μεγαλύτερο ή ίσον από
==	Ίσο
!=	Διάφορο

Προσοχή !!! == και = είναι δύο διαφορετικοί τελεστές

Λογικοί Τελεστές

- Δρουν επι ενός ή δύο τελεστέων
- Έχουν σαν βάση την δίτιμη άλγεβρα Boole
- Οι είσοδοι και έξοδοι μπορούν να λάβουν μόνο δύο τιμές TRUE και FALSE

Τελεστής	**Δράση**
&&	Λογικό AND
||	Λογικό OR
!	Λογικό NOT

x	**y**	**x&&y**	**x||y**	**!x**
T	T	T	T	F
T	F	F	T	
F	T	F	T	T
F	F	F	F	

π.χ (10+5)<(12+8) →15<20 → **TRUE**

π.χ (10>5)||(-8>10) → TRUE || FASLE →**TRUE**

Λογικοί Τελεστές

- **Στη C μια πρόταση με λογικούς τελεστές μπορεί να είναι ή αληθής ή ψευδής (TRUE,FALSE)**
  - **Η τιμή 0 θεωρείται ισοδύναμη με το ψεύδος (FALSE)**
  - **Κάθε άλλη τιμή θεωρείται ισοδύναμη με αλήθεια (TRUE)**

Νέο Πρόγραμμα

- Δημιουργήστε ένα νέο project με τίτλο

LogicalOperators

Γράψτε τον παρακάτω κώδικα:

```
int x, y, z ;
x = 0;
y = 10;
.
……
z = (x>y);
printf( " x>y is %d\n " , z );
z = (x==y);
printf( " x==y is %d\n " , z );
z = (x!=y);
printf( " x!=y is %d\n " , z );
z = (x && y);
printf( " x && y is %d\n " , z );
z = !(x && y) || (x || y);
printf( " !(x && y) || (x || y) is %d\n " , z );
```

Μετατροπές τύπων

- Όταν έχουμε τελεστέους διαφορετικών τύπων αυτοί μετατρέπονται σε ενιαίο
  - Με αυτόματο τρόπο (Υπονοούμενη)
  - Με ρητή εντολή του προγραμματιστή

Υπονοούμενη Μετατροπή Τύπων

- 3.0+ 1 / 2 ➔ 3.0 και όχι 3.5

- Όταν έχουμε δύο τύπους δεδομένων, ο στενότερος μετατρέπεται στον ευρύτερο χωρίς απώλεια

- Οι τύποι της γλώσσας ταξινομούνται ανάλογα με το μέγεθος μνήμης που απαιτούν για αποθήκευση

  char < int < long < float < double

- Στην γλώσσα C οι αριθμητικές εκφράσεις μετατρέπουν αυτόματα τον τύπο char σε int και το float σε double

Ρητή Μετατροπή

- Μετατροπή μιας τιμής σε διαφορετικό τύπο κατόπιν εντολής του προγραμματιστή
- Η διαδικασία ονομάζεται προσαρμογή
- Ο τελεστής μετατροπής είναι μοναδιαίος και έχει την μορφή (τύπος δεδομένων)
- Τοποθετείται μπροστά από μια έκφραση για να μετατρέψει την τιμή της στον περικλειόμενο σε παρενθέσεις τύπο

5 / 2 ➔ 2

(float)5 / (float)2 ➔ 2.5

Τελεστής sizeof

- Επιστρέφει τον αριθμό των bytes που η τιμή της έκφρασης ή ο τύπος δεδομένων καταλαμβάνει στη μνήμη
- Ενεργεί επάνω σε εκφράσεις ή μεταβλητές

  sizeof(x+y) sizeof(x)

  Το σύστημα δεν υπολογίζει την τιμή της έκφρασης
- Και σε τύπο δεδομένων πχ. sizeof(int)

```
int x,y;
printf( " x=%d ", sizeof( x ) );          // επιστρέφει x=4
printf( " \n y=%d ", sizeof( y ) );       // επιστρέφει x=4
printf( " \n x+y=%d ", sizeof( x+y ) );   // επιστρέφει x+y = 4
```

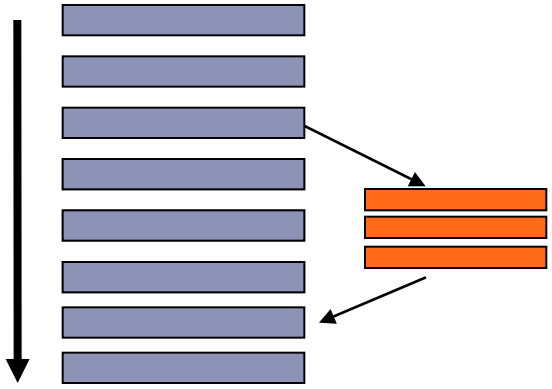
Δομημένη Ανάπτυξη Προγραμμάτων

- **Πρόγραμμα Η/Υ**

  ➔ Η επίλυση ενός προβλήματος σε Η/Υ με ένα σύνολο από προτάσεις (εντολές) σε μία γλώσσα υψηλού επιπέδου, οι οποίες εκτελούνται διαδοχικά.

- **Σειριακή Εκτέλεση**

  Όλες οι προτάσεις σε ένα πρόγραμμα στη C εκτελούνται σειριακά, η μία μετά την άλλη



Δομημένη Ανάπτυξη Προγραμμάτων

- Σύνθετα προβλήματα εμπεριέχουν καταστάσεις λογικών επιλογών (εάν, εφόσον, καθότι, εξέτασε, ελέγχω, βρες σε τι κατάσταση είναι) και επανάληψης (επανέλαβε, όσο ισχύει, για).

  π.χ. φωτεινός σηματοδότης, εξεύρεση μιας οδού, δήλωση μαθημάτων

  Αν το x είναι μεγαλύτερο ή ίσο με το 0 τότε τύπωσε ότι το x είναι θετικό αλλιώς τύπωσε ότι είναι αρνητικό

- **Μεταβίβαση Ελέγχου**

  μία πρόταση μπορεί να εκτελεστεί ακόμα και αν

  δεν είναι η επόμενη στη σειρά

Δομημένη Ανάπτυξη Προγραμμάτων

- Πρόταση **GoTo**

  Καθορισμός της μεταβίβασης ελέγχου σε ένα ευρύ φάσμα πιθανών προορισμών

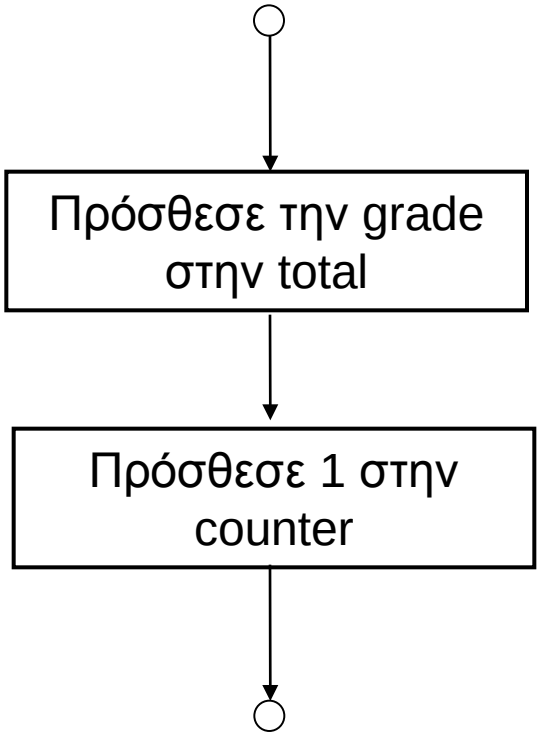
➔ **Όλα** τα προγράμματα μπορούν να γραφούν **χωρίς** προτάσεις GoTo

Μη χρησιμοποιείται σε καμία περίπτωση την εντολή GoTo. Δημιουργεί δύσκολα στην κατανόηση και στον εντοπισμό σφαλμάτων προγράμματα

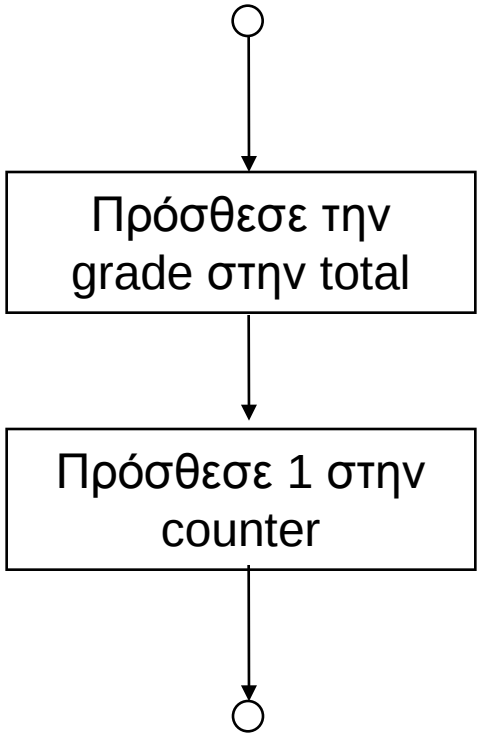
Δομημένη Ανάπτυξη Προγραμμάτων

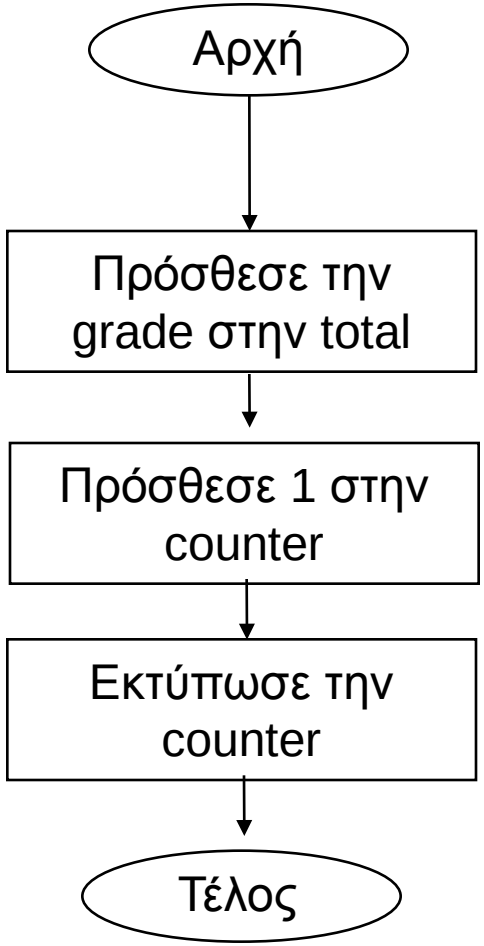
Όλα τα προγράμματα μπορούν να γραφούν με μόνο τρεις δομές ελέγχου

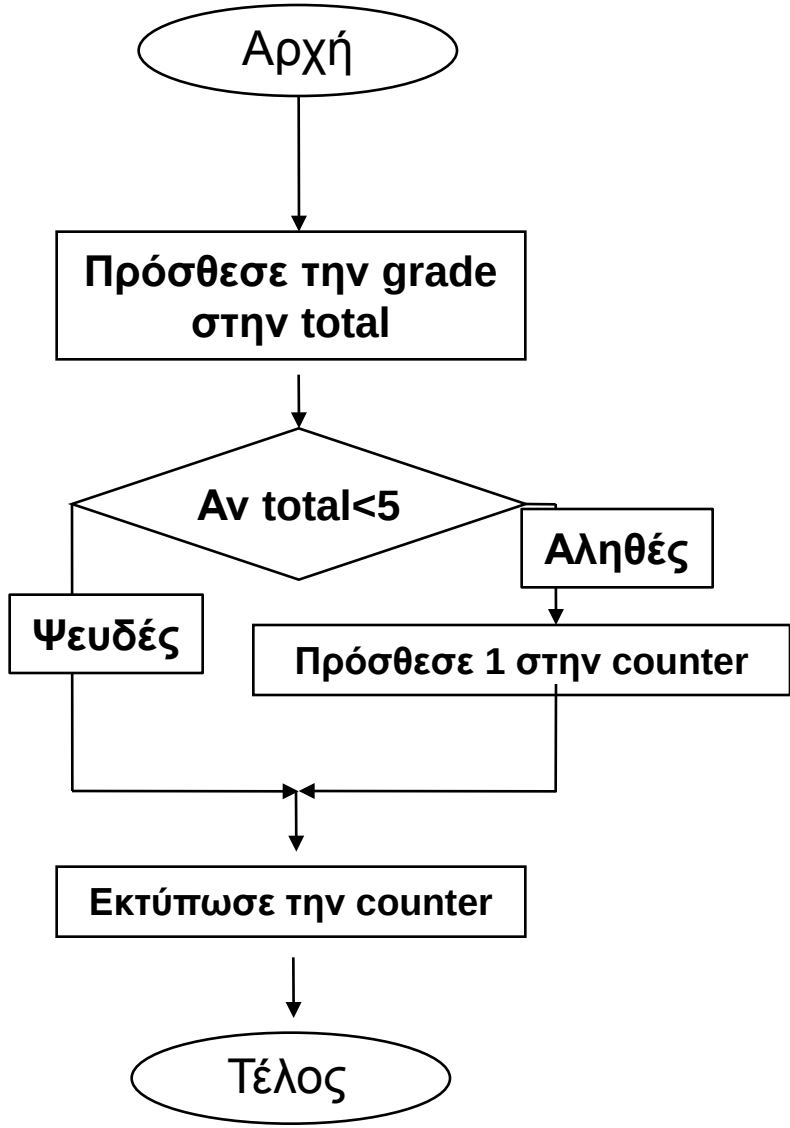
- Σειριακή δομή
- Δομή επιλογής
- Δομή επανάληψης



Διάγραμμα Ροής







Δομημένη Ανάπτυξη Προγραμμάτων

Δομές Επιλογής

- Απλή δομή επιλογής if (δομή μίας επιλογής)

```
Εάν (ισχύει αυτό)
εκτέλεσε τις (0,1,….N) εντολές
```

- Δομή επιλογής if/else (δομή διπλής επιλογής)

```
Εάν (ισχύει αυτό)
εκτέλεσε τις (0,1,….N) εντολές
Αλλιώς
εκτέλεσε τις (0,1,….N) εντολές
```

- Δομής επιλογής switch (δομή πολλαπλών επιλογών)

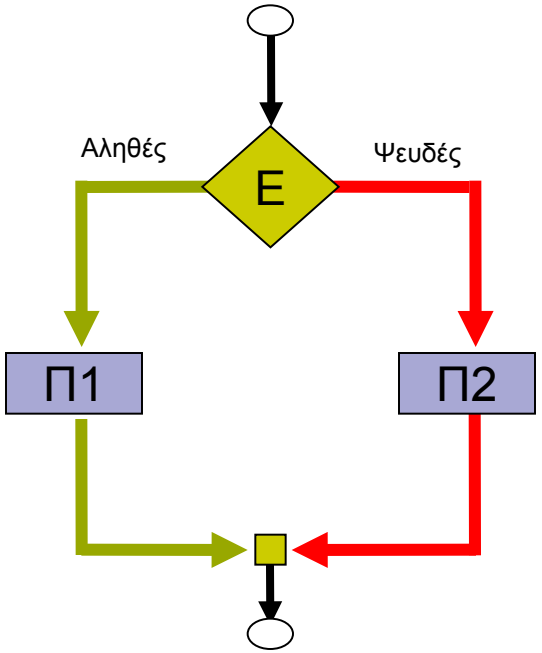
```
Εάν (η μεταβλητή)
      είναι ίση με την περίπτωση 1, εκτέλεσε τις (0,1,….N) εντολές
      είναι ίση με την περίπτωση 2, εκτέλεσε τις (0,1,….N) εντολές
      είναι ίση με την περίπτωση 3, εκτέλεσε τις (0,1,….N) εντολές
```

Δομημένη Ανάπτυξη Προγραμμάτων

- Δομές Επανάληψης
  - Δομή επανάληψης **while**
  - Δομή επανάληψης **do-while**
  - Δομή επανάληψης **for**

Υπο συνθήκη διακλάδωση (Δομή Επιλογής)

- Επιλεκτική εκτέλεση προτάσεων (η εκτέλεση προτάσεων ανάλογα με την ορθότητα ή ψεύδους μιας λογικής έκφρασης)
- Λογική έκφραση = εκφράσεις τελεστέων με λογικούς τελεστές και τελεστές συσχέτισης
- Πρόταση διακλάδωσης έχει την μορφή
  if E then Π1 else Π2
- Σε κάθε περίπτωση ο έλεγχος μεταφέρεται στο ένα και μοναδικό σημείο εξόδου



Σύνταξη της εντολής if

```
if (συνθήκη)
{
        Κώδικας που ισχύει όταν η συνθήκη είναι αληθής
}
else
{
        Κώδικας που ισχύει όταν η συνθήκη είναι ψευδής
}
```

Το κομμάτι του else είναι προαιρετικό

Τα άγκιστρα είναι προαιρετικά αν ό κώδικας που περικλείεται σε αυτά είναι μια μόνη εντολή

```
if (x>=0)
{
        printf("To x einai thetiko\n");
}
else
{
        printf("To x einai arnhtiko\n");
}
```

```
if (x>=0)
        printf("To x einai thetiko\n");
else
        printf("To x einai arnhtiko\n");
```

Νέο Πρόγραμμα

Δημιουργήστε ένα νέο project με τίτλο

if

Χρήση της IF

Να γραφεί πρόγραμμα το οποίο θα υπολογίζει την παράσταση (α+β)/γ και να τυπώνει το αποτέλεσμα στην οθόνη.
(οι τιμές των α, β, γ θα δίνονται από τον χρήστη).

Το πρόγραμμα πρέπει να λαμβάνει υπόψη του την περίπτωση που ισχύει γ=0 και να τυπώνει το μήνυμα:
“Den ginetai diairesh me to 0!”

Σημαντικά Λάθη με την if

Παράδειγμα:

```
if (x==0)
{
    printf(“De ginetai diairesh me to 0”);
}
apotelesma = 2/x;
printf(“apotelesma = %d”,apotelesma);
```

Το παραπάνω πρόγραμμα θα βγάλει μεν το μήνυμα ότι δε γίνεται 0 ΑΛΛΑ ΜΕΤΑ **ΘΑ ΣΥΝΕΧΙΣΕΙ ΝΑ ΕΚΤΕΛΕΙΤΑΙ ΚΑΙ ΘΑ ΚΑΝΕΙ ΤΗ ΔΙΑΙΡΕΣΗ!!!**

Η σωστή αντιμετώπιση είναι η χρήση else

```
if (x==0)
{
     printf(“De ginetai diairesh me to 0”);
}
else
{
    apotelesma = 2/x;
    printf(“apotelesma = %d”,apotelesma);
}
```

Νέο Πρόγραμμα

Δημιουργήστε ένα νέο project με τίτλο

if2

Χρήση της IF και Λογικών Τελεστών

Δίνεται το ακόλουθο τμήμα κώδικα:

```
int x, y;
scanf("%d",&x);
scanf("%d",&y);
if (  (x/3 == 0)  &&  ( (x*y) == (x+1) )  )
 printf("TEI");
```

Να δοθούν τέτοιες τιμές στα x και y έτσι ώστε να τυπώνεται η λέξη TEI.

Να αιτιολογηθεί η επιλογή των x και y.