Τελεστές, αριθμητικές και λογικές εκφράσεις

- **Τελεστής εκχώρησης τιμής:** `=`

  π.χ. `c = 5;  x = y = z = 0.0;`

- **Αριθμητικοί τελεστές:** `+, -, *, /, %`, τελεστής προσήμου **–**

  π.χ.
  ```
  c = a + 3;  w = w – 2*x;  y = -w;  f = 12.0/3.0;
  i = 5/4;  /* akeraia diairesh: 5/4 --> 1 */
  rem = 5%4 /* ypoloipo akeraias diaireshs */
  ```

- **Οι τελεστές μοναδιαίας αύξησης και μείωσης:** `++, --`

  π.χ.
  ```
  i++; j--; /* i = i+1; j = j-1; */
  j = i++;  /* j = i; i = i+1; */
  j = ++i;  /* i = i+1; j = i; */
  ```

- **Συνδυαστικοί τελεστές εκχώρησης:** `+=, -=, *=, /=, %=`

  π.χ.
  ```
  x += a;       /* x = x + a */
  y -= z + 1;   /* y = y – (z + 1); */
  a %= 3;       /* a = a % 3; */
  ```

Αριθμητικές Εκφράσεις

- Μια έκφραση αποτελείται από συνδυασμό τελεστών και τελεστέων
- Παραδείγματα εκφράσεων:

  α) `-6` β) `a * (b + c / d) / 20` γ) `q = 5 * 2`

  δ) `x = ++q % 3` ε) `q > 3` στ) `6 + (c = 3 + 8)`

  ζ) `x = 30 – (2 * (10 – (7 % 4)))`

  η) `y = 1 + 3 / 5 – 2 * 10 – 7 % 4`

- Για την αποτίμηση μιας αριθμητικής έκφρασης χρειάζεται να γνωρίζουμε τις προτεραιότητες των τελεστών.

Προτεραιότητες τελεστών (κατά φθίνουσα προτεραιότητα)

Τελεστές	Προσεταιριστικότητα
`() [] . -> ++ --` (postfix)	από αριστερά προς τα δεξιά
`-` (μον.πλην) `++ --` (prefix) `! (type) * & sizeof`	από δεξιά προς τα αριστερά
`*` (πολ/σμός) `/ %`	από αριστερά προς τα δεξιά
`+ -`	από αριστερά προς τα δεξιά
`< <= > >=`	από αριστερά προς τα δεξιά
`== !=`	από αριστερά προς τα δεξιά
`= += -= %= *= /=`	από δεξιά προς τα αριστερά

Σχεσιακοί Τελεστές

Μορφή έκφρασης: **τελεστέος1 τελεστής τελεστέος2**

Οι σχεσιακοί τελεστές χρησιμοποιούνται για συγκρίσεις και οι εκφράσεις αποτιμούνται σε True (αληθές ή ισοδύναμα 1) ή False (ψευδές ή ισοδύναμα 0)

- **Ισότητα**: `==` (είναι οι δύο τελεστέοι ίσοι?)
- **Μεγαλύτερο ή μικρότερο από**: `>, <`
- **Μεγαλύτερο ή ίσο**: `>=`
- **Μικρότερο ή ίσο**: `<=`
- **Ανισότητα**: `!=`

Παραδείγματα

```
1 == 2      (false)
1 != 2      (true)
x+1 > x-5   (true)
```

Λογικοί Τελεστές και Λογικές Εκφράσεις

- Τελεστές: `&&` **(and)**, `||` **(or)**, `!` **(not)**
- Λογικές εκφράσεις:

έκφραση1 `&&` **έκφραση2**	(*αληθής (true) αν και μόνο αν και οι δύο εκφράσεις είναι αληθείς*)
έκφραση1 `||` **έκφραση2**	(*αληθής αν τουλάχιστον μία από τις δύο εκφράσεις είναι αληθής*)
`!`**έκφραση**	(*αληθής αν η έκφραση είναι ψευδής (false) και αντίστροφα*)

- Σειρά αποτίμησης: από αριστερά προς τα δεξιά. Η αποτίμηση σταματάει όταν εξασφαλισθεί η ΑΛΗΘΕΙΑ ή το ΨΕΥΔΟΣ της έκφρασης.
- Παραδείγματα:

`6 > 4 && 3 == 3`	*αληθής*
`!(6 > 3) || (x = 2)`	*αληθής (γίνεται εκχώρηση τιμής)*
`(6 > 3) || (x = 2)`	*αληθής (δεν γίνεται εκχώρηση τιμής)*

Η εντολή if

- Σύνταξη μιας εντολής if :

```
if (συνθήκη)  /* Η συνθήκη είναι λογική έκφραση */
{
   εντολή1;
   εντολή2;
   . . .
   εντολήN;
}
```

Το σώμα της if αποτελείται από ένα σύνολο εντολών οι οποίες θα εκτελεστούν μόνο αν η συνθήκη της if είναι αληθής

- Παράδειγμα1

```
if (x <= 10  &&  x > 5)
{
   a = b + 2;
   y = x;
}
```

- Παράδειγμα2

```
if (x <= 10  &&  x > 5);
   y = x;
```

Προσοχή! Όχι ';'

εκτελείται πάντα

Πρόγραμμα που έμφανίζει αποτελέσματα συγκρίσεων.

```
#include <stdio.h>
int main()
{
  int  x, y;

  printf("Input two integers x and y: \n");
  scanf("%d%d",&x,&y);

  /* Compare x and y and show the result */

  if (x == y)
     printf("x is equal to y\n");

  if (x > y)
     printf("x is greater than y\n");

  if (x < y)
     printf("x is less than y\n");

  return 0;
}
```

```
Input two integers x and y:  25  42 <enter>
x is less than y
```

Η εντολή if - else

- Σύνταξη μιας εντολής if - else:

```
if (συνθήκη)
   εντολή1;   /* απλή εντολή ή σώμα εντολών {. . .} */
else
   εντολή2;   /* απλή εντολή ή σώμα εντολών {. . .} */
```

- Παράδειγμα υπολογισμού απόλυτης τιμής

```
. . .
int  x, y;
. . .
scanf("%d",&x);

if (x < 0)
   y = -x;
else
   y = x;

printf("ABS(%d) = %d\n",x,y);
. . .
```

Εμφωλευμένες if

- Παράδειγμα εμφωλευμένης if

```
if (συνθήκη1)
   εντολή1;
else
   if (συνθήκη2)
      εντολή2;
   else
      εντολή3;
```

- Παράδειγμα βελτίωσης του προγράμματος σύγκρισης των x, y

```
. . .
if (x == y)
   printf("x is equal to y\n");
else if (x > y)
        printf("x is greater than y\n");
     else
        printf("x is less than y\n");
. . .
```

Εμφωλευμένες if

- Άλλο παράδειγμα εμφωλευμένων if με ισοδύναμη επίλυση του προηγούμενου προβλήματος σύγκρισης των x, y

```
if (x >= y)
  if (x > y)
     printf("x is greater than y\n");
  else
     printf("x is equal to y\n");
else
   printf("x is less than y\n");
```

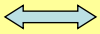
- Γενικά, η κάθε else αντιστοιχίζεται με την πλησιέστερη if εκτός και αν η if απομονώνεται σε σώμα εντολών, όπως στο παρακάτω παράδειγμα:

```
if (x >= y)
{
   if (x > y)
      printf("x-y: positive difference\n");
}
else
   printf("x-y: negative difference\n");
```

Ο τριαδικός υπό συνθήκη τελεστής ? :

- Αντιστοιχεί στην εντολή if - else:

```
if (συνθήκη)
   εντολή1;
else
   εντολή2;
```



```
συνθήκη ? έκφραση1 : έκφραση2
```

Πρώτα αποτιμάται η συνθήκη. Αν είναι αληθής τότε αποτιμάται η έκφραση1 ειδάλλως η έκφραση2.

- Παραδείγματα

```
y = (x < 0) ? -x : x;   /* y = abs(x) */

mx = (y > z) ? y : z;   /* mx = max(y,z) */

printf("Exete %d antikeimen%c.\n",n, (n==1) ? 'o' : 'a');

(b == 10) ? printf("Ten\n") : printf("Other than ten\n");

int_flag ? scanf("%d",&i) : scanf("%lf",&x);
```

Πολλαπλές επιλογές: η εντολή switch

- **Σύνταξη της switch**

η έκφραση πρέπει να αποτιμάται σε ακέραιο, δηλαδή να είναι απαριθμητού τύπου.

```
switch (έκφραση)
{
   case τιμή1:  εντολές1;  break;
   case τιμή2:  εντολές2;  break;
   . . .
   default:     εντολές;
}
```

***break**: προαιρετική εντολή που τερματίζει τη switch.*

- Παράδειγμα

```
...
char letter;    ...    scanf("%c",&letter);
switch (letter)
{
  case 'a':
  case 'e':
  case 'i':
  case 'o':
  case 'u': printf("Letter %c is a vowel!\n",letter);  break;
  default:  printf("Letter %c is a consonant!\n",letter);
}
```