«Προγραμματισμός Ι»

3-4 - Άλλες Εντολές Ελέγχου της C

Δρ. Χαράλαμπος Καρανίκας
ΠΣ Μηχανικών Πληροφορικής

Είσοδος Χαρακτήρων

Είσοδος χαρακτήρων από το πληκτρολόγιο

- Η getchar() χρησιμοποιεί line buffer και αναμένει το Enter για να επιστρέψει τον χαρακτήρα.

  char ch;
  ch = getchar();
  printf(" you typed: %c", ch);
- Η getche() χωρίς temp line character buffer επιστρέφει αμέσως τον χαρακτήρα.(conio.h)

- Κωδικοί χαρακτήρων ASCII μια διατεταγμένη ακολουθία χαρακτήρων.

  'a' μικρότερο από το 'b'

  ch = getchar();
  if (ch < 'f') printf("char is less than f");

```
/* Χρήση λειτουργιών εισόδου χαρακτήρα για την εισαγωγή μιας επιλογής από ένα μενού*/
#include <stdio.h>
int main(void)
{
  int a, b;
  char ch;

  printf("Do you want to:\n");
  printf("Add, Subtract, Multiply, or Divide?\n");
  printf("Enter first letter: ");
  ch = getchar();
  printf("\n");

  printf("Enter first number: ");
  scanf("%d", &a);
  printf("Enter second number: ");
  scanf("%d", &b);

  if(ch=='A') printf("%d", a+b);
  if(ch=='S') printf("%d", a-b);
  if(ch=='M') printf("%d", a*b);
  if(ch=='D' && b!=0) printf("%d", a/b);
  return 0;
}
```

ΈΝΘΕΣΗ ΕΝΤΟΛΩΝ if

• Όταν μία εντολή if είναι ο προορισμός μίας άλλης if ή else

```
if (count > max) /* εξωτερική if */
        if (error) printf("Error, try again."); /*ένθετη if*/
```

•Ερώτημα, ποιά if σχετίζεται με ποιά else:

```
if(p)
   if(q) printf("a and b are true");
else printf("To which if does this else apply?");
```

Απάντηση: Μία else σχετίζεται πάντα με την πλησιέστερη if του ίδιου τμήματος κώδικα και δεν είναι ήδη συσχετισμένη με μία άλλη else.(Δεύτερη if)

•Κλίμακα if-else-if:

```
if(έκφραση)
else if(έκφραση) εντολή;
else if(έκφραση) εντολή;
…….
else εντολή;
```

```
#include <stdio.h>

int main(void)
{
  int a, b;
  char ch;

  printf("Do you want to:\n");
  printf("Add, Subtract, Multiply, or Divide?\n");
  printf("Enter first letter: ");
  ch = getchar();
  printf("\n");

  printf("Enter first number: ");
  scanf("%d", &a);
  printf("Enter second number: ");
  scanf("%d", &b);

  if(ch=='A') printf("%d", a+b);
  else if(ch=='S') printf("%d", a-b);
  else if(ch=='M') printf("%d", a*b);
  else if(ch=='D' && b!=0) printf("%d", a/b);

  return 0;
}
```

΄ΕΝΘΕΣΗ ΕΝΤΟΛΩΝ if - 2

```
#include <stdio.h>

int main(void)
{
  int answer, count;
  int again;

  for(count=1; count<11; count++) {
       printf("What is %d + %d? ", count, count);
       scanf("%d", &answer);
       if(answer == count+count) printf("Right!\n");
       else {
            printf("Sorry, you're wrong\n");
            printf("Try again.\n ");

            printf("\nWhat is %d + %d? ", count, count);
            scanf("%d", &answer);


            if(answer == count+count) printf("Right!\n");
            else
                printf("Wrong, the answer is %d\n", count+count);
       }
  }
  return 0;
}
```

ΟΙ ΠΑΡΑΛΛΑΓΕΣ ΤΟΥ ΒΡΟΧΟΥ for

```
/*Συνθήκη ελέγχου*/
    #include <stdio.h>
    int main(void)
    {
      int i;
      char ch = 'a'; /* give ch an initial value */

      for(i=0; ch != 'q'; i++) {
          printf("pass: %d\n", i);
          ch = getchar();
      }
      return 0;
    }
/*Κενό προορισμό*/
    #include <stdio.h>
    int main(void)
    {
      char ch;

      for (ch=getchar(); ch!='q'; ch=getchar());
      printf("Found the q.");

      return 0;
    }
```

```
/*Τμήμα αρχικοποίησης κενό*/
    #include <stdio.h>
    int main(void)
    {
      int i;

      printf("Enter an integer: ");
      scanf("%d", &i);

      for(; i; i--) printf("%d ", i);

      return 0;
    }
/*Ατέρμων βρόχος*/
    for( ; ; ) {
      …….
    }
/*Τμήμα αύξησης*/
    for(i=0; i<10; ) {
        printf("%d ", i);
        i++;
      }
```

Ο ΒΡΟΧΟΣ while ΤΗΣ C

- while (έκφραση) εντολή;
- while (έκφραση) {τμήμα κώδικα}

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

int main(void)
{
  char ch;

  ch = getche();

  while(ch!='q') ch = getche();
  printf("Found the q.");

  return 0;
}
```

```
/*Απλή μηχανή κωδικοποίησης*/
#include <stdio.h>
#include <conio.h>

int main(void)
{
  char ch;

  printf("Enter your message.\n");

  ch = getche();
  while(ch != '\r') {
    printf("%c", ch+1);
    ch = getche();
  }
  return 0;
}
```

Ο ΒΡΟΧΟΣ do ΤΗΣ C

- do {
  εντολές;
  } while (έκφραση);

- Οι εντολές προορισμού εκτελούνται τουλάχιστον μια φορά. Για αυτό καθίσταται ιδανικός για τον έλεγχο των επιλογών που κάνει από ένα μενού ο χρήστης.

```
/*Μία λιγότερη κλήση στη συνάρτηση
getche() από το ισοδύναμο με βρόχο while*/
#include <stdio.h>
#include <conio.h>

int main(void)
{
  char ch;

  do {
    ch = getche();
  } while(ch!='q');

  printf("Found the q.");

  return 0;
}
```

Ο ΒΡΟΧΟΣ do ΤΗΣ C - 2

```
#include <stdio.h>
int main(void)
{
  int a, b;
  char ch;

  printf("Do you want to:\n");
  printf("Add, Subtract, Multiply, or Divide?\n");

 do {
    printf("Enter first letter: ");
    ch = getchar();
  } while(ch!='A' && ch!='S' && ch!='M' && ch!='D');
  printf("\n");

  printf("Enter first number: ");   scanf("%d", &a);
  printf("Enter second number: ");   scanf("%d", &b);

  if(ch=='A') printf("%d", a+b);
  else if(ch=='S') printf("%d", a-b);
  else if(ch=='M') printf("%d", a*b);
  else if(ch=='D' && b!=0) printf("%d", a/b);

  return 0;
}
```

ΧΡΗΣΗ ΤΗΣ break ΓΙΑ ΤΗΝ ΕΞΟΔΟ ΑΠΟ ΕΝΑΝ ΒΡΟΧΟ

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

int main(void)
{
  int i;
  char ch;

  /* display all numbers which are multiples of 6 */
  for(i=1; i<10000; i++) {
    if(!(i%6)) {
      printf("%d, more? (Y/N)", i);
      ch = getche();
      if(ch=='N') break; /* stop the loop */
      printf("\n");
    }
  }

  return 0;
}
```

```
/*Τερματισμός μόνο του πιο εσωτερικού βρόχου*/
#include <stdio.h>

int main(void)
{
  int i, j;

  for(i=0; i<5; i++) {
    for(j=0; j<100; j++) {
      printf("%d", j);
      if(j==5) break;
    }
    printf("\n");
  }

  return 0;
}
```

ΠΟΤΕ ΠΡΕΠΕΙ ΝΑ ΧΡΗΣΙΜΟΠΟΙΕΙΤΕ ΤΗΝ continue?

```
#include <stdio.h>

int main(void)
{
  int x;

  for(x=0; x<100; x++) {
    continue;
    printf("%d ", x); /* this is never executed */
  }

  return 0;
}
```

```
#include <stdio.h>
int main(void)
{
  int total, i, j;

  total = 0;
  do {
    printf("Enter next number (0 to stop): ");
    scanf("%d", &i);
    printf("Enter number again: ");
    scanf("%d", &j);
    if(i != j) {
      printf("Mismatch\n");
      continue;
    }
    total = total + i;
  } while(i);

  printf("Total is %d\n", total);
  return 0;
}
```

ΠΟΛΛΑΠΛΕΣ ΕΝΑΛΛΑΚΤΙΚΕΣ ΕΠΙΛΟΓΕΣ ΜΕ ΤΗΝ switch

```
#include <stdio.h>

int main(void)
{
  int i;

  printf("Enter a number between 1 and 4: ");
  scanf("%d", &i);

  switch(i) {
    case 1:
      printf("one");
      break;
    case 2:
      printf("two");
      break;
    case 3:
      printf("three");
      break;
    case 4:
      printf("four");
      break;
    default:
      printf("Unrecognized Number");
  }

  return 0;
}
```

```
#include <stdio.h>
#include <conio.h>
/*Χωρίς break εκτελούνται όλα τα υπόλοιπα cases*/
int main(void)
{
  char ch;

  do {
    printf("\nEnter a character, q to quit: ");
    ch = getche();
    printf("\n");

    switch(ch) {
      case 'a':
        printf("Now is ");
      case 'b':
        printf("the time ");
      case 'c':
        printf("for all good men");
        break;
      case 'd':
        printf("The summer ");
      case 'e':
        printf("soldier ");
    }
  } while(ch != 'q');

  return 0;
}
```

Η ΕΝΤΟΛΗ goto

- Εντολή για «άνευ όρων» μεταφορά της ροής εκτέλεσης ενός προγράμματος
- Γρήγορη εκτέλεση – Δυσνόητη δομή
- Η μετάβαση επιτρέπεται μόνο μέσα σε μία συνάρτηση

```
goto mylabel;
printf("This will not print.");
mylabel: printf("This will print.");
```

```
#include <stdio.h>

int main(void)
{
  int i;

  i = 1;
  again:
    printf("%d ", i);
    i++;
    if(i<10) goto again;

  return 0;
}
```

ΑΣΚΗΣΗ ΠΑΡΑΔΕΙΓΜΑ: Δένδρο με αστεράκια

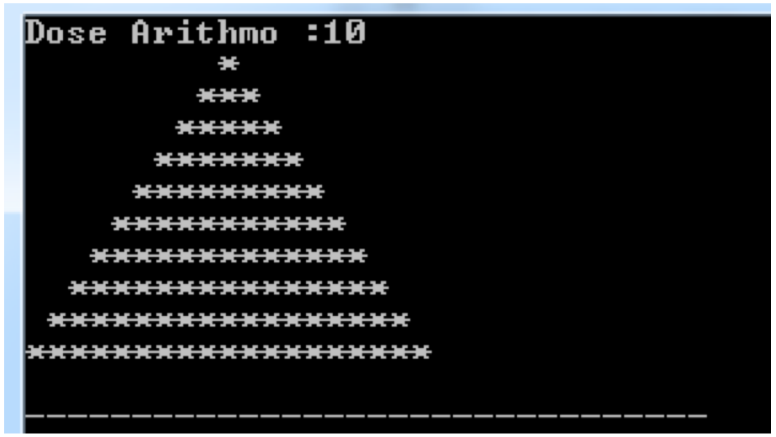
Να φτιάξετε πρόγραμμα που να διαβάζει έναν ακέραιο αριθμό Ν και να εμφανίζει στην οθόνη ένα δέντρο με Ν γραμμές από αστεράκια. Π.χ. αν Ν=5 θα πρέπει να εμφανίζεται το παρακάτω σχήμα:

```
    *
   * * *
  * * * * *
 * * * * * * *
* * * * * * * * *
```

Παρατήρηση: Η ι-οστή γραμμή έχει Ν-ι κενά και 2*ι-1 αστεράκια

ΑΣΚΗΣΗ ΠΑΡΑΔΕΙΓΜΑ: Δένδρο με αστεράκια

```
/* H.Karanikas
   Dendro me N grammes apo asterakia
*/
#include <stdio.h>


int main(void)
{
	int N,k,a,i;

	//Eisagwgi arithmoy grammwn apo asterakia
	printf("Dose Arithmo :");
	scanf("%d", &N);


	if (N==0)
		printf("No Tree...!");
	else
		for (i =1; i<=N; i++) {		//Gia kathe grammi
			for (k=1; k<=N-i; k++)			//N-i kena
				printf(" ");
			for (a=1; a<=(2*i-1); a++)		//2*i-1 Asterakia
				printf("*");
			printf("\n");
		}

	return 0;
}
```



ΕΠΑΝΑΛΑΜΒΑΝΟΜΕΝΕΣ ΚΛΗΣΕΙΣ ΣΥΝΑΡΤΗΣΕΩΝ

```
#include <stdio.h>

void recurse(int i);

int main(void)
{
  recurse(0);

  return 0;
}

void recurse(int i)
{
  if(i<10) {
    recurse(i+1); /* recursive call */
    printf("%d ", i);
  }
}
```

```
#include <stdio.h>

void recurse(int i);

int main(void)
{
  recurse(0);

  return 0;
}

void recurse(int i)
{
  if(i<10) {
    printf("%d ", i);
    recurse(i+1);
  }
}
```

Υπολογισμός του παραγοντικού

• Ορισμός του n!
n! = n x (n - 1) x … x 2 x 1

• Ο παραπάνω ορισμός μπορεί να γραφεί ως

$$n! = \begin{cases} 1 & \text{αν } n = 0 \\ n \times (n-1)! & \text{αλλιώς} \end{cases}$$

Στον εναλλακτικό ορισμό, ο υπολογισμός του n! (σύνθετο πρόβλημα) ανάγεται στον υπολογισμό του (n-1)! (απλούστερο πρόβλημα).

Αναδρομή

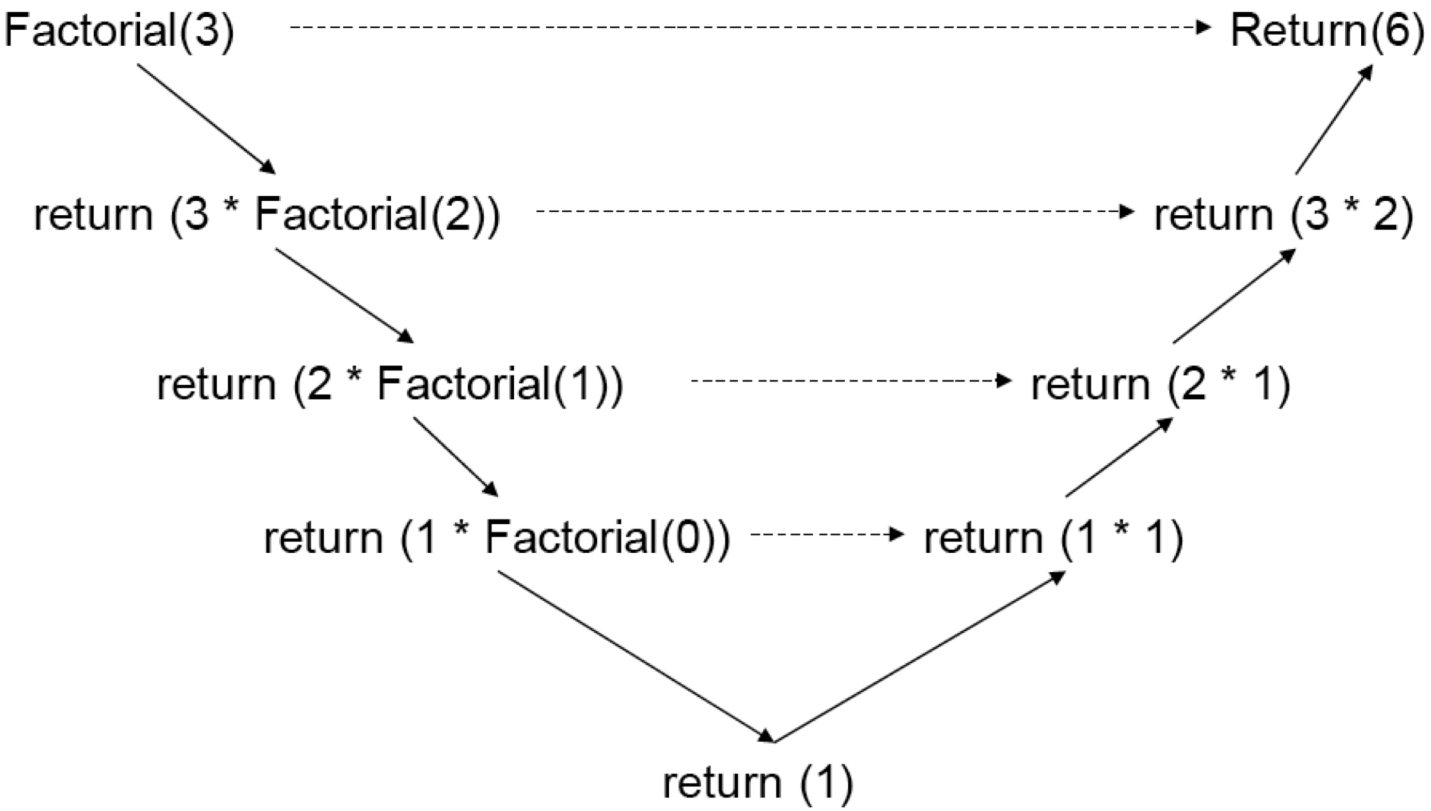
- Στρατηγική επίλυσης προβλημάτων
- Τεχνική προγραμματισμού σύμφωνα με την οποία ένα σύνθετο πρόβλημα ανάγεται σε ένα απλούστερο **της ίδιας μορφής**.
- Είναι μια *παράξενη*, μη *διαισθητική τεχνική* η οποία στην αρχή δυσκολεύει τους προγραμματιστές
- Σε κάθε περίπτωση όπου χρησιμοποιείται αναδρομή, είναι δυνατόν, εναλλακτικά, να χρησιμοποιηθεί επανάληψη
- Η χρήση της αναδρομής προσφέρει, σε κάποιες περιπτώσεις, απλούστερες λύσεις.

ΑΣΚΗΣΗ ΠΑΡΑΔΕΙΓΜΑ: Factorial

```
#include <stdio.h>

int Factorial(int N);

int main(void)
{
		int n;

		printf("Enter number");
		scanf("%d",&n);

		printf("To paragontiko to %d einai = %d\n",n, Factorial(n));

		return 0;
}

int Factorial(int N)
{

		if (N==0 || N==1)
				return 1;
		else
				return Factorial(N-1)*N;
}
```

Η αναδρομή στη C πραγματοποιείται ορίζοντας συναρτήσεις που καλούν **τον εαυτό τους** (άμεση αναδρομή)

Παράδειγμα κλήσης της Factorial



Ένα παράδειγμα: Συνάρτηση ύψωσης στη δύναμη

```
static int RaiseIntToPower(int n, int k)
{
   /* Έλεγχος απλής περίπτωσης */
   if (k ==0) {
      return (1);
   } else {
   /* Αναδρομική κλήση της ίδιας συνάρτησης */
      return (n * RaiseIntToPower(n, k-1));
   }
}
```