ΕΙΣΑΓΩΓΗ ΣΤΗ ΓΛΩΣΣΑ C ΜΕΣΩ ΠΑΡΑΔΕΙΓΜΑΤΩΝ

Θοδωρής Ανδρόνικος

Ιόνιο Πανεπιστήμιο
Τμήμα Πληροφορικής
2007-2008

Το πρώτο πρόγραμμα: Hello World

```
#include <stdio.h>

int main()
{

   printf("This is the dreaded Hello World program!\n" );

}


//  ΣΤΗΝ ΟΘΟΝΗ ΕΜΦΑΝΙΖΕΤΑΙ:
//  This is the dreaded Hello World program!
```

Μια παραλλαγή του Hello World

```
#include <stdio.h>

int main()
{

   printf( "This \n" );
   printf( "is \n" );
   printf( "the \n" );
   printf( "boring \n" );
   printf( "Hello World!\n" );

}


/*  ΣΤΗΝ ΟΘΟΝΗ ΕΜΦΑΝΙΖΕΤΑΙ:
This
is
the
boring
Hello World!
*/
```

Πρόσθεση & Πολλαπλασιασμός ακεραίων

```
#include <stdio.h>

int main()
{
   int integer1, integer2, sum, product;   /* variable declaration */

   printf( "Enter first integer\n" );
   scanf( "%d", &integer1 );               /* read first integer */
   printf( "Enter second integer\n" );
   scanf( "%d", &integer2 );               /* read second integer */
   sum = integer1 + integer2;              /* compute sum */
   product = integer1 * integer2;          /* compute product */
   printf( "Sum = %d\n", sum );            /* print sum */
   printf( "Product = %d\n", product );    /* print product */
}

/* ΣΤΗΝ ΟΘΟΝΗ ΕΜΦΑΝΙΖΕΤΑΙ:
Enter first integer
128
Enter second integer
256
Sum = 384
Product = 32768
*/
```

Ένα απλό αυτόματο

```
#include <stdio.h>
#define maxStringLength 72

int main()
{
     int i = 0, state = 0;
     char c, inputString[maxStringLength],
     runString[maxStringLength];

     printf("This program simulates a simple deterministic
     automaton.\n");
     runString[i] = state + 48;
/* The ASCII code for 0 is 48 */
     printf("Give the input string (at most %d capital letters).\n",
     maxStringLength);
     while ( ( c = getchar() ) != '\n' ) {
        inputString[ i++ ] = c;
        switch ( state ) {
           case 0:
                if ( c == 'A' || c=='a' ) state = 1; break;
            case 1:
                if ( c == `B' || c==`b' ) state = 2; break;
            case 2:
                if ( c == `C' || c==`c' ) state = 3; break;
     }
```

Ένα απλό αυτόματο (συνέχεια)

```
/* If the input string has length n, then the run of the automaton has length
    n+1 */
            runString [ i ] = state + 48;
    }
    inputString[ i ] = '\0';
    printf("You gave an input string of length %d \n", i);
    puts( inputString );
    runString [ ++i ] = '\0';
    printf("The run (length %d) of the automaton for this input was \n", i);
    puts( runString );
    if ( state == 3 )
            printf("The run was SUCCESSFUL\n");
    else
            printf("The run was UNSUCCESSFUL\n");
    return 0;
}
```

/* ΣΤΗΝ ΟΘΟΝΗ ΕΜΦΑΝΙΖΕΤΑΙ:
This program simulates a simple deterministic automaton.
Give the input string (at most 72 capital letters).
albhcf
You gave an input string of length 9
albhcf
The run (length 10) of the automaton for this input was
0112233
The run was SUCCESSFUL

Press any key to proceed
***/**

Ακέραιοι & Πραγματικοί

```
#include <stdio.h>
#define maxStringLength 72
#define Max_Counter 10
#define A 5
#define B A*A+2

void main()
{
   int counter, length, number, total, average;
   float average_1;
   double average_2, average_3;

   total = average = 0;
   average_1 = 0.0;
   average_2 = average_3 = 0.0;
   counter = 1;

   printf( "\nThe value of B is: %3.d\n\n", B );
   printf( "Give the length of the sequence (maximum 10): " );
   scanf( "%d", &length );

   while ( counter <= length ) {
      printf( "Enter number %2.d: ", counter );
      scanf( "%d", &number );
      total = total + number;
      counter = counter + 1;
   }
```

Ακέραιοι & Πραγματικοί (συνέχεια)

```
average = total / length;
   average_1 = (float) total/length;
   average_2 = total / length;
   average_3 = (double) total/length;

   printf( "The average is %2.d\n", average );
   printf( "A more accurate average is:\t\t %f \t %.10f \t %e\n",
      average_1, average_1, average_1 );
   printf( "A less accurate average is:\t\t %lf\n", average_2 );
   printf( "An even more accurate average is:\t %lf \t %.10lf \t %E\n",
      average_3, average_3, average_3 );

}
```

```
/*  ΣΤΗΝ ΟΘΟΝΗ ΕΜΦΑΝΙΖΕΤΑΙ:
The value of B is:  27

Give the length of the sequence (maximum 10): 3
Enter number  1: 3
Enter number  2: 3
Enter number  3: 4
The average is  3
A more accurate average is:                3.333333
     3.3333332539    3.333333e+000
A less accurate average is:                3.000000
An even more accurate average is:          3.333333
     3.3333333333    3.333333E+000
*/
```

Αναδρομικές Συναρτήσεις

```
// One of the simplest recursive functions: The Fibonacci function
#include <stdio.h>
#include <conio.h>

long fib (long);        // This is the function PROTOTYPE

int main()
{
    long n, fib_num;

    printf("Give a natural number: ");
    scanf("%ld", &n);
    fib_num=fib(n);
    printf("Fibonacci(%ld) = %8ld\n", n, fib_num);
    printf("Press any key to continue");
    getch();
    return 0;
}

long fib (long k)       // This is the function DEFINITION
{
    if (k==0 || k==1)
            return k;
    else
            return fib(k-1)+fib(k-2);
}

/* ΣΤΗΝ ΟΘΟΝΗ ΕΜΦΑΝΙΖΕΤΑΙ:
Give a natural number: 38
Fibonacci(38) = 39088169
Press any key to continue
*/
```

Συναρτήσεις με Πίνακες

```
#include <stdio.h>
#include <conio.h>  // Because we want to use the getch() function
#define Max_Dim 10

Initialize_Vector (int k, int factor, int X[])
{
     int i;
     for (i=0; i<k; i++)
               X[i]=i*factor*k;
}

Calc_Vector (int k, int X[], int Y[], int Z[])
{
     int i;

     for (i=0; i<k; i++)
               Z[i]=X[i]+Y[i];
}

Print_Vector (int k, int Y[])
{
     int i;
     for (i=0; i<k; i++)
               printf("The value of element %3d is: %3d\n", i, Y[i]);
}

main()
{
     int i, n, A[Max_Dim], B[Max_Dim], C[Max_Dim];
```

Συναρτήσεις με Πίνακες (συνέχεια)

```
    printf("Give the vector dimension (at most %3d): ", Max_Dim);
    do
            scanf("%d", &n);
    while (n<0 || n>Max_Dim);
    Initialize_Vector(n, 1, A);
    Initialize_Vector(n, 2, B);
    Calc_Vector(n, A, B, C);
    printf("Vector A\n");
    Print_Vector(n, A);
    printf("Vector B\n");
    Print_Vector(n, B);
    printf("Vector C\n");
    Print_Vector(n, C);
    printf("Press any key to continue\n");
    getch();                // With getchar() I must also press <ENTER>,
                            // but with getch() this is not necessary!
}

/*  ΣΤΗΝ ΟΘΟΝΗ ΕΜΦΑΝΙΖΕΤΑΙ:
Give the vector dimension (at most  10): 5
Vector A
The value of element   0 is:   0
The value of element   1 is:   5
The value of element   2 is:  10
The value of element   3 is:  15
The value of element   4 is:  20
Vector B
The value of element   0 is:   0
The value of element   1 is:  10
The value of element   2 is:  20
The value of element   3 is:  30
The value of element   4 is:  40
Vector C
The value of element   0 is:   0
The value of element   1 is:  15
The value of element   2 is:  30
The value of element   3 is:  45
The value of element   4 is:  60
Press any key to continue
*/
```

Κομψότερες Συναρτήσεις

```
#include <stdio.h>
#include <conio.h>  // Because we want to use the getch() function
#define Max_Dim 10

Initialize_Vector (int, int, int []);
Calc_Vector (int, int [], int [], int []);
Print_Vector (int, int []);

main()
{
     int i, n, A[Max_Dim], B[Max_Dim], C[Max_Dim];

     printf("Give the vector dimension (at most %3d): ", Max_Dim);
     do
          scanf("%d", &n);
     while (n<0 || n>Max_Dim);
     Initialize_Vector(n, 1, A);
     Initialize_Vector(n, 2, B);
     Calc_Vector(n, A, B, C);
     printf("Vector A\n");
     Print_Vector(n, A);
     printf("Vector B\n");
     Print_Vector(n, B);
     printf("Vector C\n");
     Print_Vector(n, C);
     printf("Press any key to continue\n");
     getch();          //          With getchar() I must also press <ENTER>,
                       //          but with getch() this is not necessary!
}
```

Κομψότερες Συναρτήσεις (συνέχεια)

```
Initialize_Vector (int k, int factor, int X[])
{
	int i;
	for (i=0; i<k; i++)
		X[i]=i*factor*k;
}
Calc_Vector (int k, int X[], int Y[], int Z[])
{
	int i;

	for (i=0; i<k; i++)
		Z[i]=X[i]+Y[i];
}
Print_Vector (int k, int Y[])
{
	int i;
	for (i=0; i<k; i++)
		printf("The value of element %3d is: %3d\n", i, Y[i]);
}

/*   ΣΤΗΝ ΟΘΟΝΗ ΕΜΦΑΝΙΖΕΤΑΙ:
Vector A
The value of element   0 is:   0
The value of element   1 is:   8
The value of element   2 is:  16
The value of element   3 is:  24
The value of element   4 is:  32
Vector B
The value of element   0 is:   0
The value of element   1 is:  16
The value of element   2 is:  32
The value of element   3 is:  48
The value of element   4 is:  64
Vector C
The value of element   0 is:   0
The value of element   1 is:  24
The value of element   2 is:  48
The value of element   3 is:  72
The value of element   4 is:  96
Press any key to continue
*/
```

Δείκτες αντί για Πίνακες

```
#include <stdio.h>
#include <conio.h>  // Because we want to use the getch() function
#include <malloc.h> // Because we want to use the malloc function

Initialize_Vector (int, int, int *);
Calc_Vector (int, int *, int *, int *);
Print_Vector (int, int *);

main()
{
      int n, *A, *B, *C;
      printf("Give the vector dimension: ");
      scanf("%d", &n);
      A=(int *)malloc(n);
      B=(int *)malloc(n);
      C=(int *)malloc(n);
      if (A!=NULL && B!=NULL && C!=NULL){
                Initialize_Vector(n, 1, A);
                Initialize_Vector(n, 2, B);
                Calc_Vector(n, A, B, C);
                printf("Vector A\n");
                Print_Vector(n, A);
                printf("Vector B\n");
                Print_Vector(n, B);
                printf("Vector C\n");
                Print_Vector(n, C);
                printf("Press any key to continue\n");
                getch();                        }
}

Initialize_Vector (int k, int factor, int *X)
{
      int i;
      for (i=0; i<k; i++)
                X[i]=i*factor*k;
}
Calc_Vector (int k, int *X, int *Y, int *Z)
{
      int i;

      for (i=0; i<k; i++)
                Z[i]=X[i]+Y[i];
}
```

Δείκτες αντί για Πίνακες (συνέχεια)

```
Print_Vector (int k, int *Y)
{
      int i;
      for (i=0; i<k; i++)
                printf("The value of element %3d is: %3d\n", i, Y[i]);
}

/*    ΣΤΗΝ ΟΘΟΝΗ ΕΜΦΑΝΙΖΕΤΑΙ:
Give the vector dimension: 12
Vector A
The value of element   0 is:   0
The value of element   1 is:  12
The value of element   2 is:  24
The value of element   3 is:  36
The value of element   4 is:  48
The value of element   5 is:  60
The value of element   6 is:  72
The value of element   7 is:  84
The value of element   8 is:  96
The value of element   9 is: 108
The value of element  10 is: 120
The value of element  11 is: 132
Vector B
The value of element   0 is:   0
The value of element   1 is:  24
The value of element   2 is:  48
The value of element   3 is:  72
The value of element   4 is:  96
The value of element   5 is: 120
The value of element   6 is: 144
The value of element   7 is: 168
The value of element   8 is: 192
The value of element   9 is: 216
The value of element  10 is: 240
The value of element  11 is: 264
Vector C
The value of element   0 is:   0
The value of element   1 is:  36
The value of element   2 is:  72
The value of element   3 is: 108
The value of element   4 is: 144
The value of element   5 is: 180
The value of element   6 is: 216
The value of element   7 is: 252
The value of element   8 is: 288
The value of element   9 is: 324
The value of element  10 is: 360
The value of element  11 is: 396
Press any key to continue
*/
```

Λίστες

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

struct listNode {                       // Για τις ΔΥΝΑΜΙΚΕΣ ΔΟΜΕΣ
  int data;                             // πρέπει να χρησιμοποιήσω
  struct listNode *nextPtr;             // self-referential structures
};
typedef struct listNode ListNode;
typedef ListNode *ListNodePtr;
void printList(ListNodePtr);
int main()
{
      int number;
      ListNodePtr headPtr = NULL, newPtr, currentPtr, prevPtr;
      puts("Give list node - 0 to exit!");
      scanf("%d", &number);
      while (number!=0) {
            newPtr = (ListNodePtr)malloc(sizeof(ListNode));
            (*newPtr).data = number;
            // I can also write newPtr->data = number;
            newPtr->nextPtr = NULL;
            prevPtr = NULL;
            currentPtr = headPtr;
            while (currentPtr!=NULL && number>=currentPtr->data) {
                  prevPtr=currentPtr;
                  currentPtr=currentPtr->nextPtr;
            }
            if (prevPtr == NULL) {
            // Αυτό σημαίνει ότι ο νέος κόμβος θα μπεί στην ΑΡΧΉ!
                  newPtr->nextPtr = headPtr;
                  headPtr = newPtr;
            }
            else {
                  prevPtr->nextPtr = newPtr;
                  newPtr->nextPtr = currentPtr;
            }
            puts("Give list node - 0 to exit!");
            scanf("%d", &number);
      };
```

Λίστες (συνέχεια)

```
        printf("%d\n", headPtr);
        printList(headPtr);
        return 0;
}
void printList(ListNodePtr headPtr)
{
        if ( headPtr == NULL )
                printf( "The List is empty! \n" );
        else {
                printf( "The list is:\n" );
                while ( headPtr != NULL ) {
                        printf( "%d --> ", headPtr->data );
                        headPtr = headPtr->nextPtr;
                }
                printf( "NULL\n" );
        }
}

/*   ΣΤΗΝ ΟΘΟΝΗ ΕΜΦΑΝΙΖΕΤΑΙ:
Give list node - 0 to exit!
2
Give list node - 0 to exit!
4
Give list node - 0 to exit!
8
Give list node - 0 to exit!
6
Give list node - 0 to exit!
-2
Give list node - 0 to exit!
-8
Give list node - 0 to exit!
-5
Give list node - 0 to exit!
0
7867760
The list is:
-8 --> -5 --> -2 --> 2 --> 4 --> 6 --> 8 --> NULL
*/
```

Value & Reference

```
#include <stdio.h>
#include <conio.h>  // Because we want to use the getch() function

int Function_1 (int);
void Function_2 (int *);
void Wrong_Swap (int, int);
void Swap (int *, int *);

int main()
{
    int number=10, number_1=1000, number_2=4000;
    printf("\nThe Initial value of number is : %3d\n", number);
    printf("\nFunction_1 returns the value : %3d\n", Function_1(number));
    printf("\nThe value of number after Function_1 is : %3d\n", number);
    Function_2(&number);
    printf("\nThe value of number after Function_2 is : %3d\n", number);
    printf("\nThe Initial value of number 1 is : %4d  and the initial value of
    number 2 is : %4d\n", number_1, number_2);
    Wrong_Swap(number_1, number_2);
    printf("\nAfter Wrong_Swap the value of number 1 is : %4d  and the
    value of number 2 is : %4d\n", number_1, number_2);
    Swap(&number_1, &number_2);
    printf("\nAfter Swap the value of number 1 is : %4d  and the value of
    number 2 is : %4d\n", number_1, number_2);
    printf("Press any key to continue\n");
    getch();
    }
int Function_1 (int number)
{
    number = 2*number*number+5;
    return number;
}
```

Value & Reference (συνέχεια)

```
void Function_2 (int *number)
{
     *number = 8* *number * *number+8;
}
void Wrong_Swap (int num_1, int num_2)
{
     int temp=num_1;

     num_1=num_2;
     num_2=temp;
}
void Swap (int *num_1, int *num_2)
{
     int temp=*num_1;

     *num_1=*num_2;
     *num_2=temp;
}
/*   ΣΤΗΝ ΟΘΟΝΗ ΕΜΦΑΝΙΖΕΤΑΙ:
The Initial value of number is :  10
Function_1 returns the value : 205
The value of number after Function_1 is :  10
The value of number after Function_2 is : 808
The Initial value of number 1 is : 1000  and the initial value of number 2 is :
     4000
After Wrong_Swap the value of number 1 is : 1000  and the value of number
     2 is : 4000
After Swap the value of number 1 is : 4000  and the value of number 2 is :
     1000
Press any key to continue
*/
```

Πίνακες

```
#include <stdio.h>
#include <conio.h>  // Because we want to use the getch() function
#define DIMENSION 10
#define X_DIM 5
#define Y_DIM 4
void By_Reference(int, int []);
void By_Value(int, const int []);
void Print_Vector (int, const int []);
void Print_Array (int, const int [][Y_DIM]); // ΕΙΝΑΙ ΑΠΑΡΑΙΤΗΤΟ ΤΟ Y_DIM
int main()
{
    int Pinakas[DIMENSION]={0, 1, 2, 3, 4},
            Array[X_DIM][Y_DIM]={{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};
    printf("\nThe Initial values Of PINAKAS Are:\n-----------------------------
    ---\n");
    Print_Vector(DIMENSION, Pinakas);
    By_Reference(DIMENSION, Pinakas);
    // ΜΕ ΤΟΝ ΤΡΟΠΟ ΑΥΤΟ, ΥΛΟΠΟΙΩ ΚΛΗΣΗ ΚΑΤΑ ΑΝΑΦΟΡΑ
    // ΚΑΙ ΜΠΟΡΩ ΝΑ ΑΛΛΑΞΩ ΤΙΣ ΤΙΜΕΣ ΤΟΥ ΠΙΝΑΚΑ
    printf("\n\nThe Modified values Of PINAKAS now Are:\n-------------------
    ------------------\n");
    Print_Vector(DIMENSION, Pinakas);
    // ΑΝ ΘΕΛΩ ΝΑ ΑΠΑΓΟΡΕΥΣΩ ΤΗΝ ΑΛΛΑΓΗ ΤΩΝ ΣΤΟΙΧΕΙΩΝ ΤΟΥ ΠΙΝΑΚΑ
    // ΧΡΗΣΙΜΟΠΟΙΩ ΤΗ ΔΗΛΩΣΗ const int []
    // ΟΠΟΤΕ ΚΑΘΕ ΠΡΟΣΠΑΘΕΙΑ ΤΡΟΠΟΠΟΙΗΣΗΣ ΟΔΗΓΕΙ ΣΕ ΣΥΝΤΑΚΤΙΚΟ
    ΛΑΘΟΣ
    printf("\nThe values Of ARRAY Are:\n----------------------\n");
    Print_Array(X_DIM, Array);
    printf("Press any key to continue\n");
    getch();
}
```

Πίνακες (συνέχεια)

```
void By_Reference(int dim, int A[])
{
    int i;
    for (i=0; i<dim; i++)
        A[i]=DIMENSION-i;
}
void By_Value(int dim, const int A[])
{
    int i;
    for (i=0; i<dim; i++)
        A[i]+=1000;
    // ΣΤΟ ΣΗΜΕΙΟ ΑΥΤΟ Ο COMPILER ΒΓΑΖΕΙ ΛΑΘΟΣ!
}
void Print_Vector (int dim, const int A[])
{
    int i;
    for (i=0; i<dim; i++)
        printf("The value of element %3d is: %3d\n", i, A[i]);
}
void Print_Array (int x_dim, const int A[][Y_DIM])
{
    int i, j;
    for (i=0; i<x_dim; i++){
        for (j=0; j<Y_DIM; j++)
            printf("%4d", A[i][j]);
        printf("\n");
    }
}
```

Πίνακες
(συνέχεια)

```
/*  ΣΤΗΝ ΟΘΟΝΗ ΕΜΦΑΝΙΖΕΤΑΙ:
The Initial values Of PINAKAS Are:
---------------------------------
The value of element   0 is:   0
The value of element   1 is:   1
The value of element   2 is:   2
The value of element   3 is:   3
The value of element   4 is:   4
The value of element   5 is:   0
The value of element   6 is:   0
The value of element   7 is:   0
The value of element   8 is:   0
The value of element   9 is:   0

The Modified values Of PINAKAS now Are:
---------------------------------------
The value of element   0 is:  10
The value of element   1 is:   9
The value of element   2 is:   8
The value of element   3 is:   7
The value of element   4 is:   6
The value of element   5 is:   5
The value of element   6 is:   4
The value of element   7 is:   3
The value of element   8 is:   2
The value of element   9 is:   1

The values Of ARRAY Are:
------------------------
   1   2   3   0
   4   5   6   0
   7   8   9   0
   0   0   0   0
   0   0   0   0
Press any key to continue

error C2166: l-value specifies const object
*/
```

Strings

```
#include <stdio.h>
#include <conio.h>  // Because we want to use the getch() function

int main()
{
     char    string_1[]="Strings in C are tricky!",
                         string_2[]={'W', 'R', 'O', 'N', 'G', '!'},
             // ΠΡΟΣΟΧΗ! ΑΥΤΟ ΕΙΝΑΙ ΛΑΘΟΣ!
                         string_3[]={'W', 'h', 'y', '?', '\0'},
             // ΔΕΝ ΞΕΧΝΑΜΕ ΤΟ '\0'
                         string_4[25];

     printf("\tGive a string : ");
     scanf("%s", string_4);
                                                  // ΕΔΩ ΠΡΕΠΕΙ ΝΑ ΛΕΙΠΕΙ ΤΟ
     &
     printf("String_1 is: %s\n", string_1);
     printf("String_2 is: %s\n", string_2);
     printf("String_3 is: %s\n", string_3);
     printf("String_4 is: %s\n", string_4);
     printf("Press any key to continue\n");
     getch();
     //       With getchar() I must also press <ENTER>,
     //       but with getch() this is not necessary!
}


/*  ΣΤΗΝ ΟΘΟΝΗ ΕΜΦΑΝΙΖΕΤΑΙ:
      Give a string : Be Careful!
String_1 is: Strings in C are tricky!
String_2 is: WRONG!¦¦¦¦¦¦¦¦¦¦Strings in C are tricky!
String_3 is: Why?
String_4 is: Be
Press any key to continue
*/
```

Dijkstra

```
#include <stdio.h>
#define n 5                             // Number of vertices
#define infinite 2147483647             // This integer represents infinitity

int Weight[n][n] = {{0, 10, infinite, infinite, 5},{infinite, 0, 1, infinite, 2},
                    {infinite, infinite, 0, 4, infinite},
                    {7, infinite, 6, 0, infinite}, {infinite, 3, 9, 2, 0}},
                   Distance[n], Parent[n], Q[n];
int Initialize(int);
int Extract_Minimum();
int Relax(int, int);
int Dijkstra(int);
int Print_Path(int, int);

int main()
{
      int source, vertex;
      printf("\n Pick the vertex that will be the source \n");
      scanf("%d", &source);
      Dijkstra(source);
      for (vertex=0; vertex<n; vertex++){
            printf("\n The shortest path from source = %d to destination =
      %d has Distance = %d \n and is the following:", source, vertex,
      Distance[vertex]);
            Print_Path(source, vertex);
            printf("\n");
      }
      return 0;
}
int Initialize(int source)
{
      int vertex;
      for (vertex=0; vertex<n; vertex++){
            Distance[vertex]=infinite;
            Parent[vertex]=-1;
            Q[vertex]=1;              // Initially, Q contains all vertices
      }
      Distance[source]=0;
      return 0;
}
```

Dijkstra (συνέχεια)

```
int Extract_Minimum()
{
	int u, vertex, min=infinite;
	for (vertex=0; vertex<n; vertex++)
		if (Q[vertex] && Distance[vertex]<min){
				min=Distance[vertex];
				u=vertex;
		}
	Q[u]=0;			// Extract u from Q
	return u;
}
int Relax(int u, int v)
{
	if (Distance[u]!=infinite && Weight[u][v]!=infinite)
		if (Distance[v]>Distance[u]+Weight[u][v]){
				Distance[v]=Distance[u]+Weight[u][v];
				Parent[v]=u;
		}
	return 0;
}
int Dijkstra(int source)
{
	int u, vertex, adjacent;
	Initialize(source);
	for (vertex=0; vertex<n-1; vertex++){
		// For the n-1 vertices excluding the source
		u=Extract_Minimum();
		for (adjacent=0; adjacent<n; adjacent++)
				Relax(u, adjacent);
	}
	return 0;
}
```

Dijkstra (συνέχεια)

```
int Print_Path(int source, int destination)
{
	if (source==destination)
		printf(" %d", source);
	else{
		Print_Path(source, Parent[destination]);
		printf(" -> %d", destination);
	}
	return 0;
}

/*============================================
	=====================================
==================================  OUTPUT
	=====================================
 Pick the vertex that will be the source
0
 The shortest path from source = 0 to destination = 0 has Distance =
	0
 and is the following: 0

 The shortest path from source = 0 to destination = 1 has Distance =
	8
 and is the following: 0 -> 4 -> 1

 The shortest path from source = 0 to destination = 2 has Distance =
	9
 and is the following: 0 -> 4 -> 1 -> 2

 The shortest path from source = 0 to destination = 3 has Distance =
	7
 and is the following: 0 -> 4 -> 3

 The shortest path from source = 0 to destination = 4 has Distance =
	5
 and is the following: 0 -> 4
============================================
	=====================================
============================================
	=====================================*/
```

sizeof

```
#include <stdio.h>
#include <conio.h>  // getch()
int main()
{
  char c;
  short s;
  int i;
  long l;
  float f;
  double d;
  long double ld;
  int array[ 20 ], *ptr = array;
  printf( "    sizeof c = %d"   "\tsizeof(char)  = %d"
        "\n   sizeof s = %d"  "\tsizeof(short) = %d"
        "\n   sizeof i = %d"   "\tsizeof(int) = %d"
        "\n   sizeof l = %d"   "\tsizeof(long) = %d"
        "\n   sizeof f = %d"   "\tsizeof(float) = %d"
        "\n   sizeof d = %d"  "\tsizeof(double) = %d"
        "\n   sizeof ld = %d"  "\tsizeof(long double) = %d"   "\n sizeof array = %d"
        "\n   sizeof ptr = %d\n",
        sizeof c, sizeof( char ), sizeof s, sizeof( short ), sizeof i, sizeof( int ), sizeof l,
        sizeof( long ), sizeof f, sizeof( float ), sizeof d, sizeof( double ), sizeof ld,
        sizeof( long double ), sizeof array, sizeof ptr );
  printf("Press any key to continue\n");
  getch();
  return 0;
}
/*  ΣΤΗΝ ΟΘΟΝΗ ΕΜΦΑΝΙΖΕΤΑΙ:
    sizeof c = 1      sizeof(char)  = 1
    sizeof s = 2      sizeof(short) = 2
    sizeof i = 4      sizeof(int) = 4
    sizeof l = 4      sizeof(long) = 4
    sizeof f = 4      sizeof(float) = 4
    sizeof d = 8      sizeof(double) = 8
   sizeof ld = 8      sizeof(long double) = 8
 sizeof array = 80
  sizeof ptr = 4
Press any key to continue
*/
```

Pointers

```
#include <stdio.h>
#include <conio.h>  // getch()
#define SIZE 4
int main()
{
      int counter, Pinakas[]={4, 8, 12, 16}, *Ptr, *Ptr2, difference;
      Ptr=&Pinakas[0];
      Ptr2=Pinakas;
      printf("\nPointer Pinakas points to address :\t\t %p", Pinakas);
      printf("\nPointer Ptr points to address :\t\t\t %p", Ptr);
      printf("\nPointer Ptr2 points to address :\t\t %p", Ptr2);
      printf("\n\nAn integer occupies %3d bytes in memory\n", sizeof(int));
      Ptr++;
      printf("\nAfter Ptr++ pointer Ptr points to address :\t %p", Ptr);
      Ptr2+=2;
      printf("\nAfter Ptr2+=2 pointer Ptr2 points to address :\t %p", Ptr2);
      Ptr2--;
      printf("\nAfter Ptr2-- pointer Ptr2 points to address :\t %p", Ptr2);
      Ptr2+=2;
      printf("\nAfter Ptr2+=2 pointer Ptr2 points to address :\t %p", Ptr2);
      difference=Ptr2-Ptr;
      printf("\nThe value of difference Ptr2-Ptr is :\t\t %d\t\tWhy?\n", difference);
      for (counter=0; counter<SIZE; counter++)
            printf("\nPinakas[%d] = %3d", counter, Pinakas[counter]);
      Ptr2=Pinakas;
      for (counter=0; counter<SIZE; counter++)
            printf("\n*(Ptr2+%d) = %3d", counter, *(Ptr2+counter));
      for (counter=0; counter<SIZE; counter++)
            printf("\n*(Pinakas+%d) = %3d", counter, *(Pinakas+counter));
      for (counter=0; counter<SIZE; counter++)
            printf("\nPtr2[%d] = %3d", counter, Ptr2[counter]);

      printf("\nPress any key to continue\n");
   getch();
   return 0;
}
```

Pointers
(συνέχεια)

```
/*
ΣΤΗΝ ΟΘΟΝΗ ΕΜΦΑΝΙΖΕΤΑΙ:
Pointer Pinakas points to address :          0012FEBC
Pointer Ptr points to address :              0012FEBC
Pointer Ptr2 points to address :             0012FEBC

An integer occupies   4 bytes in memory

After Ptr++ pointer Ptr points to address :     0012FEC0
After Ptr2+=2 pointer Ptr2 points to address :  0012FEC4
After Ptr2-- pointer Ptr2 points to address :   0012FEC0
After Ptr2+=2 pointer Ptr2 points to address :  0012FEC8
The value of difference Ptr2-Ptr is :          2          Why?

Pinakas[0] =  4
Pinakas[1] =  8
Pinakas[2] =  12
Pinakas[3] =  16
*(Ptr2+0) =  4
*(Ptr2+1) =  8
*(Ptr2+2) =  12
*(Ptr2+3) =  16
*(Pinakas+0) =  4
*(Pinakas+1) =  8
*(Pinakas+2) =  12
*(Pinakas+3) =  16
Ptr2[0] =  4
Ptr2[1] =  8
Ptr2[2] =  12
Ptr2[3] =  16
Press any key to continue
*/
```

Pointers
σε Συναρτήσεις

```
#include <stdio.h>
#include <conio.h>  // getch()
int Power2( int );
int Power3( int );
int Power4( int );
int main()
{
      int (*F_Array[ 3 ])( int ) = {Power2, Power3, Power4};
      // F_Array είναι πίνακας με 3 δείκτες σε συναρτήσεις που επιστρέφουν ακέραιες τιμές
      int i, choice, sum;
      choice=sum=0;
      for (i=0; i<10; i++)
       sum+=(*F_Array[ choice ])( i );
      printf( "\nThe sum of the integers 0 to 9 raised to power 2 is : %6d", sum);
      choice=1;              sum=0;
      for (i=0; i<10; i++)
       sum+=(*F_Array[ choice ])( i );
      printf( "\nThe sum of the integers 0 to 9 raised to power 3 is : %6d", sum);
      choice=2;              sum=0;
      for (i=0; i<10; i++)
       sum+=(*F_Array[ choice ])( i );
      printf( "\nThe sum of the integers 0 to 9 raised to power 4 is : %6d", sum);
      printf( "\nProgram execution completed.\n" );
      printf("\nPress any key to continue\n");
   getch();
   return 0;
}
int Power2( int n ){
      return n*n;
}
int Power3( int n ){
      return n*n*n;
}
int Power4( int n ){
      return n*n*n*n;
}
/*
ΣΤΗΝ ΟΘΟΝΗ ΕΜΦΑΝΙΖΕΤΑΙ:
The sum of the integers 0 to 9 raised to power 2 is :    285
The sum of the integers 0 to 9 raised to power 3 is :   2025
The sum of the integers 0 to 9 raised to power 4 is :  15333
Program execution completed.
Press any key to continue
*/
```