Παραδείγματα χρήσης συναρτήσεων

```
/*  File: scores2.c
    This program uses functions to read scores into an array and to print
    the scores.
*/
#include <stdio.h>
#define MAX 10

int read_intaray(int scores[], int lim);
void print_intaray(int scores[], int lim);
main()
{   int n, exam_scores[MAX];

    printf("***List of Exam Scores***\n\n");
    n = read_intaray(exam_scores, MAX);
    print_intaray(exam_scores, n);
}

/* Function reads scores in an array. */
int read_intaray(int scores[], int lim)
{   int n, count = 0;

    printf("Type scores, EOF to quit\n");

    while ((count < lim) && (scanf("%d", &n) != EOF)) {
         scores[count] = n;
         count++;
    }
    return count;
}

/* Function prints lim elements in the array scores. */
void print_intaray(int scores[], int lim)
{   int i;

    printf("\n***Exam Scores***\n\n");
    for (i = 0; i < lim; i++)
         printf("%d\n", scores[i]);
}
```

```
/* File: string2.c
   This program reads and writes strings until an empty string is
   read. It uses functions to read and print strings to standard
   files.
*/
#include <stdio.h>
#define SIZE 100
void print_str(char s[]);
void read_str(char s[]);

main()
{  char str[SIZE];

   do {
     read_str(str);
     print_str(str);
   } while (str[0]);
}

/* Function reads a string from standard input until a newline is
   read. A NULL is appended.
*/
void read_str(char *s)
{  int i;
   char c;

   for (i = 0; (c = getchar()) != '\n'; i++)
     s[i] = c;
   s[i] = NULL;
}

/* Function prints a string to standard output and terminates with a
   newline.
*/
void print_str(char *s)
{  int i;

   for (i = 0; s[i]; i++)
     putchar(s[i]);
   putchar('\n');
}
```

Ταξινόμηση

```
Original    34    8     64    51    32    21
After i=1
After i=2   8
After i=3   8     21
After i=4   8     21    32    34    51    64
After i=5   8     21    32    34    51    64
```

Εύρεση μικρότερου στοιχείου και της θέσης του

```
n = 10;
minInd = 0;
min = array[minInd];
for (i=0;i<n;i++)
        if (min < array[i])
        {
                min = array[i];
                minInd = i;
        }
printf("min is %d at position %d\n",min,minInd);
```

Συνάρτηση ταξινόμησης 1Δ πίνακα με straight selection sort

```
void straightselection(int array[],int length)
{
  int i,j,minInd,tmp;
  for(j=0;j<length-1;j++)
  {
    minInd=j;
    for(i=j+1;i<length;i++)
    {
      if(array[i]<array[minInd])
        min=i;
        minInd = i;
    }
    if(minInd!=j)
    {
      tmp=array[j];
      array[j]=array[minInd];
      array[minInd]=tmp;
    }
  }
```

```
}
```

Συνάρτηση ταξινόμησης 1Δ πίνακα με straight bubble sort

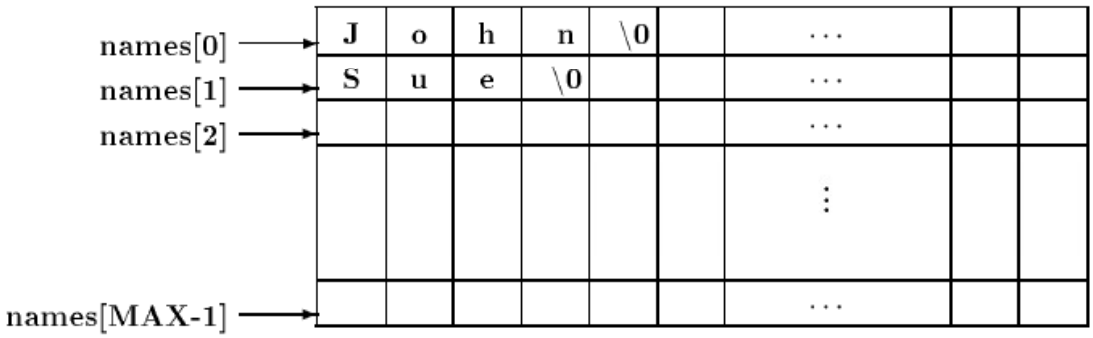
"5 1 4 2 8"

(5 1 4 2 8) σε (1 5 4 2 8)
(1 5 4 2 8) σε (1 4 5 2 8)
(1 4 5 2 8) σε (1 4 2 5 8)
(1 4 2 5 8) σε (1 4 2 5 8)

```
for (i=0; i<n-1; i++)
{
	for (j=0; j<n-1-i; j++)
		if (a[j+1] < a[j]) /* compare the two neighbors */
		{
			tmp = a[j];	/* swap a[j] and a[j+1]	*/
			a[j] = a[j+1];
			a[j+1] = tmp;
		}
}
```

Πίνακας ονομάτων

```
char names [MAX] [SIZE];
```



k = srchstrtab(names, n, key);

Εύρεση σε πίνακα από ονόματα:

The value returned is assigned to an integer variable, k. If successful, srchstrtab() returns the index where the string was found; otherwise, it returns -1.

```
* ***String Search for John Smith***
   *
   * John Smith found at index 2
```

```
/*   File: strsrch.c
          Other Source Files: strtab.c
          Header Files: strtab.h
          This program searches for a string (key) in a set of strings
          in a two dimensional array. It prints the index where key is found.
          It prints -1, if the string is not found.
     */

     #include <stdio.h>
     #define MAX 10
     #include "strtab.h"

     main()
     {    int k;
          char names[MAX][SIZE] = { "John Jones", "Sheila Smith",
                                        "John Smith", "Helen Kent"};

          printf("***String Search for John Smith***\n\n");
          k = srchstrtab(names, 4, "John Smith");
          printf("John Smith found at index %d\n", k);
     }
```

```
/*   File: strtab.c - continued */
     #include <string.h>
     /*   Searches a string table strtab[][] of size n for a string key. */
     int srchstrtab(char strtab[][SIZE], int n, char key[])
     {    int i;

          for (i = 0; i < n; i++)
               if (strcmp(strtab[i], key) == 0)
                    return i;
          return -1;
     }
```

```
/*   File: strsort.c
          Other Source Files: strtab.c
          Header Files: strtab.h
          This program sorts a set of strings in a two dimensional array.
          It prints the unsorted and the sorted set of strings.
     */

     #include <stdio.h>
     #define MAX 10
     #include "strtab.h"

     main()
     {    int n;
          char names[MAX][SIZE] = { "John Jones", "Sheila Smith",
                                        "John Smith", "Helen Kent"};

          printf("***String Array - unsorted and sorted***\n\n");
          printf("Unsorted ");
          printstrtab(names, 4);

          sortstrtab(names, 4);
          printf("Sorted ");
          printstrtab(names, 4);
     }
```

```
/*    File: strtab.c - continued */
     /*    Sorts an array of strings. The number of strings in the
           array is lim.
     */
     void sortstrtab(char strtab[][SIZE], int lim)
     {    int i, eff_size, maxpos = 0;
          char tmp[SIZE];

          for (eff_size = lim; eff_size > 1; eff_size--) {
               for (i = 0; i < eff_size; i++)

                    if (strcmp(strtab[i], strtab[maxpos]) > 0)
                         maxpos = i;

               strcpy(tmp, strtab[maxpos]);
               strcpy(strtab[maxpos], strtab[eff_size-1]);
               strcpy(strtab[eff_size - 1], tmp);
          }
     }
```

SORT:

```
* ***String Array - unsorted and sorted***
*
* Unsorted Names are:
* John Jones
* Sheila Smith
* John Smith
* Helen Kent
*
* Sorted Names are:
* Helen Kent
* John Jones
* John Smith
* Sheila Smith
```

Εύρεση σε ένα πίνακα ακολουθιακά

Finding a Data Item - The Search Problem

SRCH0: Search an array for an index where the key is located; if key is not present, print a message. Repeat until an end of file is entered for the key.

```
/*   File: sortsrch.h
          This file contains prototypes for sort and search functions.
     */
     int seqsrch(int x[], int lim, int key);
```

initialize index i to 0
    traverse the array until exhausted
        if array[i] matches key
            return i;
    return -1.

```
/*   File: sortsrch.c  */
     #include <stdio.h>
     #include "sortsrch.h"
     #define DEBUG
     /*
          Linear or sequential search of an array x[] of size lim for
          an item key.
     */
     int seqsrch(int x[], int lim, int key)
     {    int i;

          for (i = 0; i < lim; i++)
               if (x[i] == key)
                    return(i);
          return(-1);
     }
```

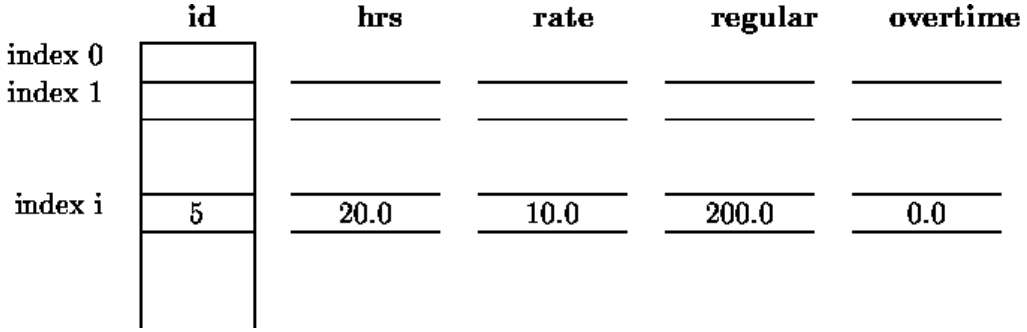
```
/* File: sortsrch.c - continued */
     /* Prints an array horizontally. */
     void pr_aray_line(int x[], int lim)
     {    int i;

          for (i = 0; i < lim; i++) {
               if (i % 10 == 0)
                    printf("\n");
               printf("%d  ", x[i]);
          }

          printf("\n");
     }

     /* File: sortsrch.h - continued */
     void pr_aray_line(int x[], int lim);
```

```
* ***Sequential Search***
*
* The id array is:
* 45 67 12 34 25 39
*
* Type an integer, EOF to quit: 23
* Item 23 is not in the array
*
* Type an integer, EOF to quit: 12
* Item 12 is found at index 2
*
* Type an integer, EOF to quit: 45
* Item 45 is found at index 0
*
```

```
/*   File: sortsrch.c - continued */
     /*   Function uses binary search to search for item in the array y[]. */
     int binsrch(int y[], int lim, int key)
     {    int low, mid, high = lim - 1;

          low = 0;

          while (low <= high) {          /* Is the array exhausted?   */
               mid = (low + high) / 2;   /* If not, find middle index */

               if (key == y[mid])        /* Is the key here?          */
                    return(mid);         /* If so, return index.      */

               else if (key < y[mid])    /* else if key is smaller,   */
                    high = mid - 1;      /* reduce the high end;      */

               else
                    low = mid + 1;       /* otherwise, increase low   */
          }
          return(-1);                    /* Not found, return -1      */
     }
```

```
* ***Binary Search***
*
* The array is:
* 12 29 30 32 35 49
*
* Type a number, EOF to quit: 12
* 12 found at array index 0
*
* Type a number, EOF to quit: 23
* 23 not found in array
*
* Type a number, EOF to quit: 34
* 34 not found in array
*
* Type a number, EOF to quit: 45
* 45 not found in array
*
* Type a number, EOF to quit: 30
* 30 found at array index 2
```

Παράδειγμα ταξονόμησης με πολλαπλούς πίνακες. Η ταξινόμηση εδώ γίνεται με βάση τον πίνακα “id”



```
/* File: payutil.c - continued */
     #include "tfdef.h"
     void sortdata(int id[], float hrs[], float rate[], int lim)
     {    int i, k, temp, swap = TRUE;
          float ftmp;

          for (i = 0; swap && i < lim - 1; i++) {
               swap = FALSE;

               for (k = 0; k < lim - i - 1; k++)
                    if (id[k] > id[k + 1]) {
                         temp = id[k];
                         id[k] = id[k + 1];
                         id[k + 1] = temp;

                         ftmp = hrs[k];
                         hrs[k] = hrs[k + 1];
                         hrs[k + 1] = ftmp;

                         ftmp = rate[k];
                         rate[k] = rate[k + 1];
                         rate[k + 1] = ftmp;
                         swap = TRUE;
                    }
          }
     }

     /* File: payutil.h - continued */
     void sortdata(int id[], float hrs[], float rate[], int lim);
```