FEEG6002 Advanced Computational Methods 1:

Laboratory-Assignment 7

Contents

Prerequisites: (revision, arrays)

1 Training: mixing strings

In a file training7.c write:

A function char*mix(char *s1, char *s2) which takes two strings s1 and s2 of equal length, and returns a new string r that mixes the characters from the input strings so that the first char in the result string r is the first char from s1, the second char in r is the first char from s2, the third char in r is the second char from s1 and the fourth char in r is the second char from s2, and so on.

The function mix needs to allocate memory for this new string r.

You can assume that the function mix is only called with two strings s1 and s2 that have equal length. There is no need to check for this in the function.

If no memory can be allocated, the function should return NULL.

Example: For input of s1="Hello" and s2="12345", the returned string should be "H1e2l3l4o5".

Here is some code you may find useful to support testing of your function mix:

```
void use_mix(void) {
   char s1[] = "Hello World";
   char s2[] = "1234567890!";

   printf("s1 = %s\n", s1);
   printf("s2 = %s\n", s2);
   printf("r  = %s\n", mix(s1, s2));
}
```

Submit your file training7.c to feeg6002@soton.ac.uk with subject line training 7 for testing and feedback.

2 Laboratory: Sorting algorithm

Given the following pseudo code of the sorting algorithm bubble sort, implement a version of this algorithm in C (source: http://en.wikipedia.org/wiki/Bubble_sort, Oct 2010) in the function void bubble(int A[], int length):

```
function bubbleSort( A )        #where A is list of sortable items
  n = length(A)-1
  for(a=0; a<= n; a++)          # C notation used for for-loop
    for(b=n; b>a; b--)          # C notation used for for-loop
      if A[b-1] > A[b] then
        swap (A[b-1], A[b])
      end if
    end for
  end for
end function
```

We recommend you expand the following template bubble_template.c which will generate random numbers for you and allow you to check whether the sorting is working. Save your working file as lab7.c.

bubble_template.c

```
#include<stdio.h>  /* provides standard input/output tools */
#include<stdlib.h> /* provides RAND_MAX                    */

#define N 20       /* number of random numbers             */

/* Pseudo code reads:
"""procedure bubbleSort( A : list of sortable items )
  n := length(A)-1
  for(a=0; a<= n; a++)
     for(b=n; b>a; b--)
        if A[b-1] > A[b] then
           swap (A[b-1], A[b])
        end if
     end for
  end for
end procedure
"""
*/

/* Given an Array A of int, use bubble sort to sort elements in A (in
   place).*/
void bubble(int A[], int length) {
  ; /* code needs to be added here */
}

/* Given an array of int 'a' of length 'length', print the first and
   last 'k' values */
void print_int_array(int a[], int length, int k) {
  int i;
  if (2*k < length) { /* longish array; only printing first and last k
                         values */
    for (i=0; i<k; i++)
```

```
      printf("a[%d]=%3d, ",i,a[i]);
    printf("   . . . .   ");
    for (i=length-k; i<length; i++)
      printf("a[%d]=%3d, ",i,a[i]);
  }
  else { /* for very short arrays, print all the data */
    for (i=0; i<length; i++ )
      printf("a[%d]=%3d, ",i,a[i]);
  }
  printf("\n");
}

int main(void) {
  int i;
  int data[N];
  /* initialises array with random integers between 0 and 999 */
  for (i=0; i<N; i++) {
    data[i] = (int) ((rand()+0.5)/(double) RAND_MAX * 999);
  }
  /* print data (at least beginning and end) */
  print_int_array(data,N,5);
  /* actual sorting: */
  bubble(data,N);
  printf("Data is now sorted:\n");
  /* print data (at least beginning and end) */
  print_int_array(data,N,5);

  return 0;
}
```

Email your file lab7.c attached to an email with subject line lab 7 to feeg6002@soton.ac.uk for feedback.