Information and Computer Science Department
First Semester 161
ICS 103 – Computer Programming in C

Final Examination

Saturday, January 14, 2017

Duration: 120 minutes

Name: KEY

Lect Serial #

ID#:

Please tick your section:

Instructor	Section
Mr. Said Abdallah Muhammad	[ ] 01 (UT 7:00 – 7:50) [ ] 02 (UT 7:00 – 7:50)
Dr. Hamood Al-Jamaan	[ ] 03 (UT 8:00 – 8:50) [ ] 04 (UT 8:00 – 8:50)
Dr. Rafiul Hassan	[ ] 05 (UT 11:00 – 11:50) [ ] 06 (UT 11:00 – 11:50) [ ] 07 (UT 13:10 – 14:00) [ ] 08 (UT 13:10 – 14:00)
Dr. Muhammad Balah	[ ] 09 (MW 7:00 – 7:50) [ ] 10 (MW 7:00 – 7:50)
Mr. Muhammad Aslam	[ ] 11 (MW 8:00 – 8:50) [ ] 12 (MW 8:00 – 8:50)
Dr. Samer Arafat	[ ] 13 (MW 9:00 – 9:50) [ ] 14 (MW 9:00 – 9:50) [ ] 15 (MW 10:00 – 10:50) [ ] 16 (MW 10:00 – 10:50)
Dr. Louai Al-Awami	[ ] 17 (MW 11:00 – 11:50) [ ] 18 (MW 11:00 – 11:50) [ ] 19 (MW 13:10 – 14:00) [ ] 20 (MW 13:10 – 14:00)

Instructions:

1. Answer all questions. Make sure your answers are clear and readable.
2. The exam is closed book and closed notes. No calculators or any helping aides are allowed. Make sure to turn off your mobile phone and keep it in your pocket.
3. If there is no space on the front of a question's page, use the back of the page. Indicate this clearly.

Question #	Maximum Grade	Obtained Grade	Remarks
1	15		
2	35		
3	15		
4	15		
5	20		
Total	100		

Question # 1 [15 points]

1. [5 points] Convert the following C if-statement into equivalent switch-statements. Assume that k and m are integer variables and that x is initialized.

```
if(k == 2|| k == 5)
     x += 2;
else{
      if(m == 8)
          x  -= 3;
      else if(m == 10)
          x  += 4;
}
```

```
switch(k){
     case 2:
     case 5: x += 2;
              break;
     default:  switch(m){
                  case 8: x -= 3;
                       break;
                  case 10:  x += 4;
               }
 }
```

2. [5 points] Convert the following program fragment into an equivalent fragment that uses while-loops instead of for-loops:

```
double product = 1 ;
int k, m;
for(k = 1; k < 16; k += 2){
  for(m = k; m <= 12; m++ )
       product *= (k + m);
  printf("p = %f\n",product);
}
```

```
double product = 1;
int k, m;
k = 1;
while(k < 16){
    m = k;
    while(m <= 12){
        product *= (k + m);
        m++;
    }
    printf("p = %f\n",product);
    k += 2;
}
```

3. [5 points] Given the following declarations:

```
int  k, m;
char  ch = 'A';
```

Write C nested loops to print the following pattern:

```
1A  2A  3A  4A  5A  6A
1B  2B  3B  4B  5B
1C  2C  3C  4C
1D  2D  3D
1E  2E
1F
```

```
for(k = 6; k >= 1; k--){
      for(m = 1; m <= k; m++ ){
        printf("%d%c ", m, ch);
      }
      ch++;
      printf("\n");
 }
```

Question # 2 [35 points]

Write the output of each of the following C programs or program fragments:

[3 points]

```c
int x = 5, y = 8;
printf("%d\n", x++ + --y);
printf("%d  %d\n", x, y);
```

Output:

```
12
6    7
```

[3 points]

```c
#include <stdio.h>
#include <ctype.h>
int main(void){
      char str[ ] = "Ab";
     tolower(str[0]);
     toupper(str[1]);
     puts(str);
     return 0;
}
```

Output:

```
Ab
```

[3 points]
What is the output if the input is:
DHAHRAN IS COLD TODAY

```c
char str1[81], str2[81];
scanf("%s", str1);
gets(str2);
printf("%s", str2);
puts(str1);
printf("%s", str1);
```

Output:

```
 IS COLD TODAYDHAHRAN
DHAHRAN
```

[5 points]

```c
int matrix[][4] = {{5, 12, 8, 1},
       {9, 5, 4, 2}, {3, 7, 1, 4},
          {11, 20, 13, 16}}, k, m;
for(m = 2; m >= 0; m--){
 for(k = 2; k >= 0; k--){
    printf("%2d ",matrix[m][k]+ matrix[k][3]);
  }
  printf("\n");
}
```

Output:

```
 5      9      4
 8      7     10
12     14      6
```

[4 points]

```c
int a = 4, b = 5, *p1, *p2;
p1 = &a;
p2 = &b;
*p2 = *p2 + 5;
b =  *p1 * *p2;
*p1 = 6;
printf("%d  %d", a, *p2 );
```

Output:

```
6     40
```

[5 points]

```
#include <stdio.h>
int funcn(int a, int* b);
int main(void) {
  int  x[] = {1, 2, 3, 4};
  printf("%d %d\n",funcn(x[1],  &x[3]), x[3]);
  printf("%d  %d\n",  x[1], x[3]);
  return 0;
}

int funcn(int a, int* b) {
      *b = 3 * *b;
      a = 2;
      return a + *b;
}
```

Output:

```
14      4
2       12
```

[7 points]

```
#include <stdio.h>
#include <strings.h>
int main(void) {
  char strings[][80]
           = {"THIS", "IS", "THE", "BEST"};
   int k, m;
   for(k = 0; k <= 3; k++){
     for(m = strlen(strings[k]) - 1; m >= 0; m--){
         printf("%c", strings[k][m]);
      }
       printf("\n");
    }
    printf("\n");
    for(k = 0; k <= 2; k++){
        strcat(strings[k], strings[k + 1]);
     }

      for(k = 0;k < 4; k++)
        puts(strings[k]);

  return 0;
}
```

Output:

```
SIHT
SI
EHT
TSEB

THISIS
ISTHE
THEBEST
BEST
```

[5 points]

```
      char str[ ] = "\n32\tRba*";
      int k;
      printf("%d\n", strlen(str));
      for(k = 0; k < strlen(str); k++){
            if( islower(str[k]))
               str[k] = toupper(str[k]);
            else if(isspace(str[k]))
               str[k] = '*';
            else if(isdigit(str[k]))
                str[k] = str[k] + 3;
            else
               str[k] = '?';
      }
      puts(str);
```

Output:

```
8
*65*?BA?
```

Question # 3 [15 points]

Write a function that receives two integer 1D-arrays **x** and **y** and the sizes of these arrays: **sizeX** and **sizeY** [Assume that each of the arrays is filled with values]

The function returns **1** if **y** contains a sub-array whose size is the same as **sizeX** and whose sum equals the sum of array **x**. If no such sub-array of **y** exists the function returns **0**.

Examples for four different calls to the function:

Array x	Array y	Returned value	Reason
{2, 1, 5}	{**3, 2, 3**, 4, 1}	1	Sum of subarray {3, 2, 3} equals sum of array x
{2, 1, 5, 3}	{3, 2, **3, 4, 1, 3**, 5}	1	Sum of subarray {3, 4, 1, 3} equals sum of array x
{4, 6, 9}	{4, 6}	0	No subarray of y of size 3 has sum equal to array x
{3, 2, 5}	{3, 2, 3, 7, 1}	0	No subarray of y of size 3 has sum equal to array x

For line 2 where x = {2, 1, 5, 3} and sumx = 11. The subarrays of **y** checked are:
{3, 2, 3, 4} with sum = 12, then {2, 3, 4, 1} with sum = 10, then {3, 4, 1, 3} with sum = 11 which is equal sumx. The function will stop here and return 1.
For the last line where x = {3, 2, 5} and sumx = 10. The subarrays of **y** checked are:
{3, 2, 3} with sum = 8, then {2, 3, 7} with sum = 12, then the last subarray is {3, 7, 1} with sum = 11. Since all subarrays sums are not equal to 10 the function returns 0.

Note: Your function must be general and it must not contain **scanf** or **printf**.
DO NOT WRITE THE MAIN FUNCTION

```
int subArrayExists(int x[], int y[], int sizeX, int sizeY){

      if(sizeX > sizeY)
            return 0;

    int i, k, sumX = 0, sumY;
    for(k = 0; k <= sizeX - 1; k++)
         sumX += x[k];

     i = 0;
     while(sizeX + i - 1 <  sizeY){
           sumY = 0;
          for(k = i; k  <= sizeX + i - 1; k++){
                sumY += y[k];
           }

           if(sumX == sumY)
              return 1;
           else
             i++;
     }

     return 0;
}
```

Question # 4 [15 points]

Write a function that receives the number of students in a class, the number of quizzes they have taken, and a 2D-array containing the student scores, where each row stores the scores of a particular student. The function returns an integer 1D-array containing the quiz numbers of the Quizzes with average above **1.5** and the size of this array.

Sample input array for a class of 6 students who have taken 6 quizzes each:

	Qz1	**Qz2**	**Qz3**	**Qz4**	**Qz5**	**Qz6**			
Student1	3.0	1.0	3.0	1.0	0.5	1.0	. . .		
Student2	2.7	1.7	3.0	1.8	1.5	1.8	. . .		
Student3	3.0	0.0	2.9	1.0	1.1	2.0	. . .		
Student4	2.5	1.2	2.8	1.2	0.0	1.2	. . .		
Student5	3.0	1.5	3.0	1.2	1.3	1.5	. . .		
Student6	2.7	1.3	3.0	1.4	0.0	1.7	. . .		
	⋮	⋮	⋮	⋮	⋮				
							. . .		
							. . .		

Since the averages of Quiz1, Quiz2, Quiz3, Quiz4, Quiz5 and Quiz6 are **2.81**, 1.11, **2.95**, 1.26, 0.73 and **1.53** respectively, the returned output array of quiz numbers will be:

1	3	6			. . .		

And the returned array size is 3

<u>**Note:**</u>
- You may assume that a constant **MAXQUIZZES** is declared before the main function.
- Your function must be general. It must work for any valid number of students and any valid number of quizzes.
- Your function must not contain **scanf** or **printf** calls.
- DO NOT WRITE THE MAIN FUNCTION.

```
void getQuizzNumbers(int numStudents, int numQuizzes,
        double scores[][MAXQUIZZES], int quizNums[], int* quizCount){
      int colmn, row, index = 0;
      double quizSum, average;
      for(colmn = 0; colmn <= numQuizzes - 1; colmn++){
            quizSum = 0;
            for(row = 0; row <= numStudents - 1; row++){
                   quizSum += scores[row][colmn];
            }
            average = quizSum / numStudents;
            if(average > 1.5){
                  quizNums[index] = colmn + 1;
                  index++;
            }
      }
      *quizCount = index;
}
```

Question # 5 [20 points]

Write a complete C program that reads a text-file **input.txt** that contains English words and writes to an output file **output.txt** all words in **input.txt** that are in uppercase followed by the number of uppercase words found. Your program must display an error message on the screen if **input.txt** does not have an uppercase word.

A word is separated from the next word by any number of the following nine characters: blank character **' '** , tab character **'\t'** , new line character **'\n'** , full stop **'.'** , comma **','** , semi-colon **';'** , colon **':'**, exclamation mark **'!'**, and question mark **'?'**.

Note:

- Your program must be general; it must work for any input text-file.
- You may assume the maximum number of characters in a line is 120
- Your program <u>**may**</u> use an **int** function **wordIsUpperCase** that receives a string and returns **1** if all the characters in the string are uppercase; otherwise it returns **0**
  [Note: A function that receives a string may also receive a pointer of type char*]

- The behavior of your program must be similar to the sample run shown below.

Sample input.txt

```
this IS
a story OF a SMALL boy, aged NINE
WHO saved the LIFE of a person.
It HAPPENED one COLD;;;DAY

In,a small VILLAGE
In the MOUNTAINS south OF here.
What????a wonderful story!
```

Sample output.txt

```
IS
OF SMALL NINE
WHO LIFE
HAPPENED COLD DAY

VILLAGE
MOUNTAINS OF
Number of uppercase words found = 12
```

// **Solution 01**: Assuming a maximum line size of 120 characters and using function

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
int  wordIsUpperCase(char st[]);
int main(void){
      FILE *infile, *outfile;
      infile = fopen("input.txt", "r");
      if(infile == NULL){
            printf("Error in opening input.txt");
            exit(1);
      }
      outfile = fopen("output.txt", "w");

      char line[121], delimeters[] = " .,;:!?\n\t";
      char *word;
      int wordCount = 0, k;
      while( fgets(line, 121, infile) != NULL){
            word = strtok(line, delimeters);
            while(word != NULL){
                   if(wordIsUpperCase(word)){
                     fprintf(outfile, "%s ", word);
                     wordCount++;
                   }

                 word = strtok(NULL, delimeters);
             }

            fprintf(outfile, "\n");
      }
      if(wordCount == 0)
         printf("Error: No uppecase word in input.txt");
    else
       fprintf(outfile,"Number of uppercase words found = %d\n",wordCount);
    fclose(infile);
    fclose(outfile);
    return 0;
}
int  wordIsUpperCase(char st[]){
int i;
for(i=0; st[i]!='\0';i++) {
 if(!isupper(st[i]))
    return 0;
}
return 1;
}
```

Solution 02 : Assuming a maximum line size of 120 characters

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>

int main(void){
      FILE *infile, *outfile;
      infile = fopen("input.txt", "r");
      if(infile == NULL){
            printf("Error in opening input.txt");
            exit(1);
      }

      outfile = fopen("output.txt", "w");

      char line[121], delimeters[] = " .,;:!?\n\t";
      char *word;
      int wordCount = 0, wordIsUpperCase, k;
      while( fgets(line, 121, infile) != NULL){
            word = strtok(line, delimeters);
            while(word != NULL){
               wordIsUpperCase = 1;
                  for(k = 0; k <= strlen(word) - 1; k++){
                       if(! isupper(word[k])){
                          wordIsUpperCase = 0;
                          break;
                        }
                }

                if(wordIsUpperCase){
                     fprintf(outfile, "%s ", word);
                     wordCount++;
                }

                word = strtok(NULL, delimeters);
             }

            fprintf(outfile, "\n");
      }

      if(wordCount == 0)
         printf("Error: No uppecase word in input.txt");
       else
        fprintf(outfile,"Number of uppercase words found = %d\n",wordCount);

       fclose(infile);
      fclose(outfile);

      return 0;
}
```

Solution 03: Solution that does not use strtok and that can accommodate any line size

```
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
int main(void){
      FILE *infile, *outfile;
      infile = fopen("input.txt", "r");
      if(infile == NULL){
            printf("Error in opening input.txt");
            exit(1);
      }

      outfile = fopen("output.txt", "w");
     int inWord = 0, k, index = 0, wordCount = 0, wordIsUpperCase;
      char ch, word[20];
      while( fscanf(infile, "%c", &ch) != EOF){
            if(isalpha(ch)){
                  inWord = 1;
                 word[index] = ch;
                 index++;
            }
            else if(inWord && ! isalpha(ch)){
                  inWord = 0;
                   wordIsUpperCase = 1;
                  for(k = 0; k < index; k++){
                     if(! isupper(word[k])){
                       wordIsUpperCase = 0;
                       break;
                 }
             }
             if(wordIsUpperCase){
                 word[index] = '\0';
                 wordCount++;
                 fprintf(outfile, "%s ", word);
             }
             index = 0;
            }

            if(ch == '\n')
                 fprintf(outfile, "\n");
      }

      if(wordCount == 0)
          printf("Error: The file has no uppercase word");
       else
         fprintf(outfile,"Number of uppercase words found = %d\n",wordCount);
      fclose(infile);
      fclose(outfile);
      return 0;
}
```