بسم الله الرحمن الرحيم

KING FAHD UNIVERSITY OF PETROLEUM AND MINERALS

Information and Computer Science Department

2010/2011 First Semester (Term 101)

ICS103 Programming in C (2-3-3)

SECOND EXAM

Thursday, December 30th 2010, 09:30 AM – 11:30 AM

120 MINUTES

Student Information

Name:	
KFUPM ID:	

EXAM INSTRUCTIONS

- Use of Calculator in this exam is NOT allowed.
- If you go outside the exam hall for any reason (including going to toilet or bathroom), you will NOT be allowed to return back to the exam.
- Cheating in the exam will result in an F grade and other serious consequences.

MLAIH	UT 07:00-7:50	
BOUCHEKHMA	SM 02:10-03:00	
RAFI UL-HASAN	SM 07:00-7:50	SM 09:00-09:50
AL-DARWISH	UT 09:00-09:50	UT 01:10-02:00
SAID	UT 08:00-08:50	UT 10:00-10:50

CIRCLE YOUR SECTION

Scored Marks

Question No.	Maximum Score	Score
01	18	
02	15	
03	9	
04	9	
05	7	
06	13	
07	13	
08	16	
TOTAL	100	

Question 1 (18 POINTS): **1.5 for Each Output Value (Format is Important)**

Find the output of the following programs:

I)

```
#include<stdio.h>
int main(void){
int  k, m, n;
n = 0;
for(k = -7; k < 13; k += 4){
   n++;
   for(m = 3; m <= 6; m += 5)
       n += 4;
   n += 2;
}
printf("%d\n%d\n%d\n",n,k,m);
return(0);}
```

OUTPUT

```
35
13
8
```

II)

```
#include<stdio.h>
int main(void){
int k = 0, x = 5, done = 0;
do{
   done = 1;
   scanf("%d",&x);
   for(k = x + 1; k > 4; k--)
       done = 0;
} while(x != 5 && !done);
printf("%d %d\n%d\n",x,k,done);
return(0);}
```

The Input for this Program is:

6 7 2 5

OUTPUT

```
2  3
1
```

III)

```
#include<stdio.h>
int main(void){
int i = 10, j = -6;
while(i < 12){
     while(-1 > j){
          printf("%d %d\n",i,j);
          j = j + 3;
     }
     i = i + 1;
}
printf("%d %d\n",i,j);
return(0);}
```

OUTPUT

```
10  -6
10  -3
12  0
```

Question 2 (15 POINTS): **1.5 for Each Output Value (Format is Important)**

Find the output of the following programs:

I)

```
#include<stdio.h>
#include<math.h>
int main(void){
int s, x;
FILE *f1, *f2;
f1 = fopen("file1.txt","r");
f2 = fopen("file2.txt","w");
for(s = fscanf(f1,"%d",&x); s != EOF; s = fscanf(f1,"%d",&x))
     if ((int) sqrt(x) % 3)
          fprintf(f2,"%d\n", x * 2);
     else
          printf("%d\n", x / 2);
fclose(f1); fclose(f2);
return(0);}
```

file1.txt

```
1    9    81
16
```

file2.txt

```
2
32
```

OUTPUT

```
4
40
```

II)

```
#include <stdio.h>
int main (){
  int x = 6, y = 4, z;
  int *p1, *p2;
  p1 = &x;
  p2 = &y;
  z = x;
  x = y;
  y = z;
  printf("%d %d\n", *p1, *p2);
  *p1 = x+3;
  *p2 = ++x – y;
  printf("%d %d\n", x, y);
  p1 = p2;
  printf("%d %d\n", *p1, *p2);
  return 0;}
```

OUTPUT

```
4    6
8    2
2    2
```

Question 3 (9 POINTS): **1.5 for Each Output Value (Format is Important)**

Find the output of the following program:

```
#include<stdio.h>

void first(double *n, double *m, double h);
void second(double *x, double *y, double z);
double third(double j, double k);

int main(void){
double a, b = 1, c;
scanf("%lf%lf",&a,&c);
first(&a,&b,c);
printf("%f %f %f\n",a,b,c);
return 0;}

void first(double *n, double *m, double h){
*n = *n – *m;
*m = h * *n;
printf("%f %f\n",*n, *m);
second(n,m,h);}

void second(double *x, double *y, double z){
z = third(*x,*y);
printf("%f\n",z);
*y = 2.0 * z;
*x = 2.0 * *y;}

double third(double j, double k){
j++;
k--;
return j + k;}
```

The Input for this Program is:

6.0 4.0

OUTPUT

5.0000	20.0000	
25.0000		
100.0000	50.0000	4.0000

Question 4 (9 POINTS): **1.5 for Each Output Value (Format is Important)**

Find the output of the following program:

```
#include<stdio.h>
int main(void){
int a[6], b[6], j;
for(j = 0; j < 6; j += 2)
   scanf("%d",&a[j]);
for(j = 5; j >= 1; j -= 2)
   scanf("%d",&a[j]);
printf("%d %d %d\n",a[1],a[3],a[4]);
for(j = 0; j < 6; j += 2)
   if (a[j] > 4){
      a[j] -= 2;
      b[j] = a[j] + 4;}
   else{
      a[j] -= 4;
      b[j] = a[j] + 3;}
printf("%d %d %d\n",b[0],b[2],b[4]);
return 0;}
```

The Input for this Program is:

4 2 7 11 -3 6

OUTPUT

```
6   -3   7

3   1   9
```

Question 5 (7 POINTS): **A statements must be in Correct Scope for Credit**

Rewrite the following program by converting all the "for" loops to their equivalent "while" loops:

```
#include<stdio.h>
   int main(void){
   double sum = 0;
   int i, j;
   for(j = 20; j <= 201; j += 10)
      for(i = -j; i <= 5; i += j/2)
         sum += i/j;
   printf("%f\n",sum);
   return 0;}
```

```
#include<stdio.h>
int main(void){
double sum = 0;
int i, j;
j = 20;                          0.5
while(j <= 201){                 1.0
   i = -j;                       1.0
   while(i <= 5){                1.0
     sum += i/j;                 1.0
     i += j/2;                   1.0
   }
   j += 10;                      1.0
}
printf("%f\n",sum);              0.5
return 0;}
```

Question 6 (13 POINTS):

Write a **RECURSIVE** integer function **odd_count** that takes a positive integer input variable **n**, and then returns the count of odd digits in **n**. As an example, if **n** = 63827, **odd_count** function should return 2, this is because there are two odd digits in **n** = 63827 (these are 3 and 7).

Note:
Write an only recursive solution for **odd_count** function, do NOT use any loop. No credit will be given to a non-recursive solution. Don't write a main function.

Hint:
Recursively test the digits of **n** one by one, if the current digit is odd, then increment and repeat the same test for the remaining digits. In case the current digit is even, then only repeat the same test for the remaining digits without incrementing. You should stop when you finish testing all digits (**n** will be shrinking for every recursive call).

```
int odd_count(int n){                                  Header  3.0
  if (n == 0)                                  Terminating Condition 2.0
      return 0;                                                  1.0
  else if (n % 10 % 2 != 0)                      Recursive Condition 3.0
      return 1 + odd_count(n / 10);                              2.0
  else
      return odd_count(n / 10);}                                 2.0
```

Question 7 (13 POINTS):

Write a function **evndiv** that receives a positive integer **n** and returns the **count** and the **sum** of the even divisors of **n** other than **n** itself. For example if the function receives 16, it returns 3 as the **count**, and 14 as the **sum**, this is because the even divisors of 16 are 2, 4 and 8. As another example, if **evndiv** receives 9, it should return 0 as the **count** and 0 as the **sum** because there are no even divisors for 9.

```
void evndiv(int n, int *count, int *sum){              Header  5.0
int i;
*sum = 0; *count = 0;                            Initializations  1.0
for (i = 2; i <= n / 2; i += 2){                            Loop  3.0
    if (n % i == 0){                                     Condition  2.0
       *count = *count + 1;                                       1.0
       *sum = *sum + i;}                                          1.0
 }
}
```

Question 8 (16 POINTS):

Write a **main** function that opens the input file **"input.txt"** which contains unknown number of integer values. If the file can't be opened, the program should print a proper message and exit. If the file open is successful, the program should read all the integer values from the input file into integer array **x** (assume the maximum size of **x** is 100). After that, for every element in **x**, the program should find the **count** and **sum** of even divisors of the element by calling **evndiv** function referenced in the previous question (question 7). Finally, for every element in **x**, the program should print the element, the **count** and **sum** of even divisors of the element in a separate line inside output file **"output.txt"**

As an example, if the contents of input file **"input.txt"** is as follows:

input.txt

```
9

16

8
```

The program should create the following output file **"output.txt"**:

output.txt

```
9     0     0

16    3     14

8     2     6
```

Notes:

-Even if you didn't solve question 7, you can still solve this question.
-Don't forget to close all opened files.
-Write only the main function.

```
int main(void){
int i = 0, count, sum, x[100];                          Array Declaration  0.5
FILE *in, *out;                        File Descriptors Declarations  1.0
in = fopen("input.txt","r");                 Opening File for Reading  1.0
if (in == NULL){                             Checking File Open Error  3.0
   printf("Can't open input file!\n");
   exit(-1);}
out = fopen("output.txt","w");               Opening File for Writing  1.0
while( fscanf(in,"%d",&x[i]) != EOF)                    Loop till EOF  4.0
{
  evndiv(x[i],&count,&sum);                Calling evndiv function  3.0
  fprintf(out,"%d %d %d\n",x[i],count,sum);  Writing to Output File  2.5
  i++;}
return 0;}
```