King Fahd University of Petroleum and Minerals
Information and Computer Science Department
ICS 103: Computer Programming in C
Summer Semester 2009-2010 (Term-093)

Major Exam-II

Time: 120 minutes **Thursday August 12, 2010**

Name:	KEY
ID#:	

PLEASE CIRCLE YOUR SECTION BELOW:

Section	01	02	03
Instructor	Mr. AHMAD IRFAN	Dr. FARAG AZZEDIN	Dr. AIMAN EL-MALEH
Time	SUMT 9:20-10:10am	SUMT 10:30-11:20am	SUMT 10:30-11:20am

Question #	Maximum Marks	Obtained Marks
1	42	
2	10	
3	20	
4	28	
Total	100	

Notes.
1. Make sure you have **NINE** pages including the cover page.
2. Closed book and notes
3. Write clearly, briefly and precisely
4. Cheating will result in ZERO grade

Good Luck

Question 1: (42 points)

Determine the output of each of the following programs:

```
#include <stdio.h>   // P1:      6 points
#include <stdlib.h>

int main (void) {

  int j, k=0;
  do {
     for(j=0; j < abs(3-k); j++)
         printf("*");
         printf("%d\n",j);
         k++;
  } while (k <= 5);
  return 0;
}
```

***3
**2
*1
0
*1
**2

```
#include <stdio.h> // P2:        6 points
int main() {
    int a=5,b=1,c;
    int *p1,*p2;

    p1=&a;
    p2=&b;
    *p2=*p1*4;
    *p1=b*2;
    c=*p1+1;
    printf("a=%d, b=%d, c=%d\n",a,b,c);

    return 0;
}
```

a=40, b=20, c=41

```
#include <stdio.h> // P3:                    6 points
int test(int i){
   printf ("%d\n",i%10);
   if (i/10==0)
      return i;
   else
      return i%10*test(i/10);
}
int main() {
  printf("%d\n",test(2345));
  return 0;
}
```

```
5
4
3
2
120
```

```
#include <stdio.h> // P4:                    6 points
int atest(int a[], int n){
   int i, t;
   for(i=0; i<n/2; i++){
      t=a[i];
      a[i]=a[n-i-1];
      a[n-i-1]=t;
   }
}
int main() {
  int x[12]={1,2,3,4,5,6,7,8,9,10};
  atest(x,12);
  for (int i=0; i<12; i++)
     printf("%d ", x[i]);
  return 0;
}
```

```
0 0 10 9 8 7 6 5 4 3 2 1
```

```
#include <stdio.h>  // P5:        6 points
int myFun (int *a, int *b, int c);
int main(void) {
     int x=2, y=3, z=10;
     z = myFun(&x,&y, z);
     printf("%d\n%d\n%d", x, y, z);
     return 0;
}
int myFun (int *a, int *b, int c)
{
    int z;
    z = *b;
    *a = z + c;
    c = *a + *b;
    *b = c + *a;
    return z;
}
```

```
13
29
3
```

```
#include <stdio.h>  //P6:         6 points
int main() {
    int x , y, temp;
    int A[] = {6, 12, 3, 9};

    for (x=1; x<=3; x++){
      for (y=0; y<=2; y++) {
        if(A[y] > A[y+1]) {
           temp = A[y];
           A[y] = A[y+1];
           A[y+1] = temp;
        }
      }
    }
    for(x=0; x<4; x++)
       printf("%d\n", A[x]);
    return 0;
}
```

```
3
6
9
12
```

```c
#include <stdio.h>  // P7:        6 points
int main()
{
    int n[]={1,2,3,4,5,6};
    int i;
    for(i=0; i<6; i++)
       n[i]=i++;
    for(i=0; i<6;i++){
       printf("%d\t", n[i]);
       if(i%2 == 0)
          printf("\n");
    }
    return 0;
}
```

```
0
2    2
4    4
6
```

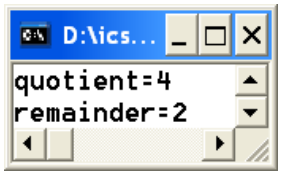
Question 2: (10 points)

Write a function **intdiv** that receives 2 input arguments for the 2 operands of the integer division i.e. numerator and denominator. The function will return the quotient and the remainder of the integer division of numerator over denominator. The function does not print anything.

```
void intdiv (int num, int den, int *q, int *r){
    *q = num/den;
    *r = num%den;
}
```

Complete the following main function so that the function **intdiv** is called to compute the quotient and remainder of dividing 18 over 4 and the results are printed as shown below. Assume that the function definition will be written after the main function.

```
#include <stdio.h>

void intdiv (int num, int den, int *q, int *r);


int main() {


  int q, r;

  intdiv (18, 4, &q, &r);

  printf("quotient=%d\n", q);
  printf("remainder=%d\n", r);

    return 0;
}
```



Question 3: (20 points)

The values of π can be determined by the series equation:

$$\pi = 4\,(1-\frac{1}{3}+\frac{1}{5}-\frac{1}{7}+\frac{1}{9}-\frac{1}{11}+\frac{1}{13}-...)$$

For example, if the number of terms is 1, π=4 and if the number of terms is 2, π=2.67.

(i) Write a function using loops, **pi_loop**, that takes the number of terms as input and returns an approximation of the value of π for the given number of terms. Write the function definition only without writing the main function.

```
double pi_loop(int n) {

double pi=0;

   for (int i=1; i<=n; i++)
       pi += 4*pow(-1,i+1)/(2*i-1);
   return pi;
}
```

(ii) Write a recursive function, **pi_recursive** that takes the number of terms as input and returns an approximation of the value of π for the given number of terms. Write the function definition only without writing the main function.

```
double pi_recursive(int n) {

if (n==1) return 4;
else return 4*pow(-1,n+1)/(2*n-1)+pi_recursive(n-1);
}
```

Question 4: (28 points)

Assume that the input file "input.txt" contains a set of points represented by their x and y coordinates. We need to write a program to compute the maximum and minimum distance among these points. Write a complete c program to do the following:

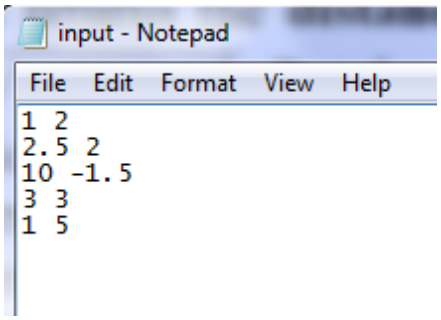
(i) Write a function, ***distance***, that takes the coordinates of two points (x1, y1) and (x2, y2) as input and returns the **distance** between them. Distance is computed as $distance = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$.

(ii) Declare two arrays XArray and YArray, of maximum size of 100 each, to hold the x and y coordinates of the points. The maximum size 100 has to be declared as a constant using define preprocessor.

(iii) Write the code fragment to read the x and y coordinates of the points from the file "input.txt" and store them in the arrays XArray and YArray. You need to determine the number of points read from the file. Your program should work for any number of points in the input file up to 100. Your program should handle file not found error.

(iv) Assuming that the x and y coordinates of the points are stored in the arrays XArray and YArray, write the code fragment to find the two points with the maximum distance and the two points with the minimum distance. The points with maximum and minimum distance are displayed along with their distances on the screen. Assume that the maximum possible distance is 1000.

Sample execution of the program is given below:




```
The maximum distance is 11.10 between the point (10.00,-1.50) and (1.00,5.00)
The minumum distance is 1.12 between the point (2.50,2.00) and (3.00,3.00)
Press any key to continue . . .
```

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#define SIZE 100

double distance(double x1, double y1, double x2, double y2);
int main() {

double XArray[SIZE], YArray[SIZE], xt, yt, dist, maxd, mind;
FILE *infile;
int i, j, maxi1, maxi2, mini1, mini2, status, count=0;
```

```
       infile = fopen("input.txt","r");
       if (infile==NULL){
         printf("Input file is not found\n");
         system("pause");
         exit(1);
       }

       status=fscanf(infile,"%lf%lf", &xt, &yt);
       while (status != EOF){
           XArray[count]=xt;
           YArray[count]=yt;
           count++;
           status=fscanf(infile,"%lf%lf", &xt, &yt);
       }

      maxd=0; mind=1000;
      for (i=0; i<count-1;i++)
        for (j=i+1; j<count; j++){
            dist =distance(XArray[i],YArray[i], XArray[j],YArray[j]);
            if  (dist>maxd){
                maxd=dist; maxi1=i; maxi2=j;
            } else if  (dist<mind){
                mind=dist; mini1=i; mini2=j;
            }
        }

      printf("The maximum distance is %.2f between the point (%.2f,%.2f) and
(%.2f,%.2f)\n", maxd, XArray[maxi1], YArray[maxi1], XArray[maxi2],
YArray[maxi2]);
      printf("The minumum distance is %.2f between the point (%.2f,%.2f) and
(%.2f,%.2f)\n", mind, XArray[mini1], YArray[mini1], XArray[mini2],
YArray[mini2]);
  fclose(infile);
  system("pause");
  return 0;
  }

  double distance(double x1, double y1, double x2, double y2){

         return sqrt(pow(x2-x1,2)+pow(y2-y1,2));
  }
```