Department of Computer Science and Engineering

Assignment 5

Subject : Programming and Data Structure (CS19003)

Instructions:

- Give the name of the programs files as <Your roll>_<assignment number>_<question number>.c. For example, 21XX12345_A1_Q1.c for question 1 of assignment 1 of a student with roll 21XX12345. **Follow the file naming convention strictly**.
- Apart from the main .c file for each program, you should also upload one additional temporary .c file for each program (such as when you have finished half of the code). The naming for the temporary file should be in the format <Your roll>_<assignment number>_<question number>_temp.c. For example, 21XX12345_A1_Q1_temp.c **Make sure that your main code do not deviate much from its temporary code for each program**.
- You should upload the main .c file and the temporary .c file individually to the Moodle course web page once you finish answering each question. No need to zip the files.
- The **deadline** to upload the programs is 12:00PM. Beyond that, submission will be closed (No extensions will be granted).
- If you do not follow the instructions, your marks may be deducted.

[Total Marks = 100 (30 + 30 + 40)]

1. Write a C program to check if the two numbers form a Zazzy pair or not. Two numbers are said to be Zazzy pair, if they satisfy the following criteria as given below:

   (a) Both the numbers should have the same number of digits and their length (number of digits) is even.

   (b) The number formed from the digits of the **first half of the first number (MSB side)** is equal to the reverse of the number formed from the second half of the digits of the second number (LSB side).

   You must use two **functions** for solving this question,

   (a) int checkDigit(int,int) - to check (i) whether two numbers have the same number of digits or not and (ii) Whether their length is even or not

   (b) int isZazzy(int,int) - to check whether the two numbers form a zazzy pair or not. **[30]**

   Example 1:

   Input:

   Number 1 = 123456

   Number 2 = 675321

   Output:

   Zazzy pair

[Reason: The number formed from the digits of the first half of the first number, i.e '123' is equal to the reverse of the number formed from the digits of the last half of the second number, i.e '321']

Example 2:

Input:

Number 1= 123456

Number 2= 321654

Output:

Not a Zazzy pair

Example 3:

Input:

Number 1 = 12345

Number 2 = 54321

Output:

Not a Zazzy pair

Hint:

To provide you with an additional help as to how to frame and formulate your code using functions for this question, consider the code snippet below. Make careful observations as to how you call the appropriate functions at the appropriate places and handle the return values, if any.

```
1   #include<stdio.h>
2
3   int checkDigit(int,int); //Function declaration
4   int isZazzy(int,int); //Function declaration
5
6   int main()
7   { //First, input the two numbers, say the input are stored in variables n and m respectively.
8    //Next proceed to compute for Zazzy pair if and only if the two numbers satisfy conditions a and b of the question.
9
10  int flag = 0;
11
12  if(checkDigit(n,m) == 1)
13  { //Now check whether the two numbers are Zazzy or not
14      if(isZazzy(n,m) == 1)
15      {
16          flag = 1;
17      }
18  else
19  {
20      flag = 0;
21  }
22  }
23
24  else
25  {
26      flag = 0;
27  } //Now if flag == 1, then numbers are Zazzy, so print accordingly, else print not Zazzy.
28  } //end of main()
29
30  int checkDigit(int n, int m)
31  { //Returns 1 if two numbers passed as parameters are of equal length and even number of digits, 0 otherwise
32  }
33
34  int isZazzy(int n, int m)
35  { //Returns 1 if n and m passed as parameters form a Zazzy pair, 0 otherwise
36  }
```

2. Consider a standard deck of cards containing 52 playing cards divided into four sets of spades, heart, diamond and clubs, with each set containing 13 cards, nine number cards with the numbers '2' to '10 printed on them and four letter cards with the letters 'A (Ace)', 'J (ack)', 'K (King)' and 'Q (Queen)' printed on them. Now for simplicity, consider the number card '10' removed from each set, so each set now contains 12 cards in total. Three siblings Alice, Bob, Sarah play a game, where five cards are randomly picked from the deck of cards and given to one of them. In the same manner, five cards are given to each of the remaining two players. The player with the highest sum of the cards wins the game. The cards J (Jack), K (King), Q (Queen), A (Ace) have values of 11, 12, 13 and 1 respectively. For winning the game, only the sum of the card values matter, their corresponding set does not matter. Write a C program following the below points to play this game.

   (a) Write a C function **int counting_cards(char arr[ ])** that computes the sum of the cards of a player as per the character array passed as argument.

   (b) Take three character arrays (char Alice[5], char Bob[5], and char Sarah[5]) to take the cards distributed to each sibling and find the winner. Your input should be either from '2' to '9' (as characters, not int) for denoting the number cards (note that card number '10' has been removed) or 'A', 'J', 'K' and 'Q' for the four letter cards, for each of the three siblings. You must call int counting_cards(char arr[]) from main() for each of the three siblings, determine the winner and display the winner in main() itself. In case there is a tie among sum values **for any two** or all of the siblings, simply print 'game tied'. **[30]**

   Example 1:

   Input:

   Alice: AJ23K

   Bob: 234Q8

   Sharah: 12678

   Output:

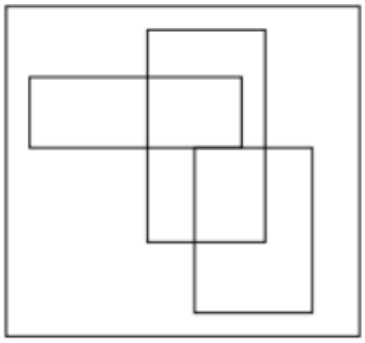
   Alice total: 29

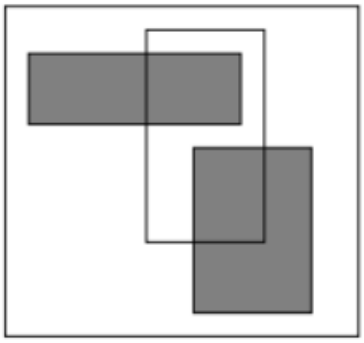
   Bob total: 30

   Sarah total: 24

   Bob wins!

3. Two rectangles are said to overlap if there exists a common point lying inside or on the boundary of both rectangles. Consider the image below, for the first figure all the rectangles overlap, while for the second figure the two shaded rectangles do not overlap. Assume that all rectangles have edges parallel to the x and y axes, and all rectangles lie completely within the first quadrant itself (i.e their corner point coordinates are all positive numbers). The rectangle can be represented by a pair of diagonal points (x1, y1) and (x2, y2), using an array rect[4] = {x1, y1, x2, y2}.





(a) Write a C function **int overlap(float [ ], float [ ])** that receives the diagonally opposite corner points of two rectangles as arguments. The function returns 1 if the rectangles overlap, otherwise 0.

(b) Write a main() program that reads the values of coordinates of diagonally opposite corner points for three rectangles r1, r2 and r3 . For coordinates, consider floating point numbers as well. Hence, for the three rectangles, consider each unique pair of rectangles and print whether the pair overlaps or not. Call the function int overlap(float [ ], float [ ]) to check for each pair. For output, simply print the rectangle pair and its overlap status. **[40]**

For example:

(r1, r2) — Overlaps

(r1, r3) — Does not overlap

(r2, r3) — Overlaps

If all the 3 pairs of rectangles overlap, then you may print (r1, r2, r3) - overalp

---