Computing bootcamp : C programming

Final exam
2024.08.26 10:00 ~ 12:00

[General instructions]

❖ **Do not include any additional header files** beyond those already provided.

❖ **Consider edge cases** on your own as part of your solution.

❖ **Complete the function in the space provided.** Do not modify any other parts of the program.

❖ **Submit the provided skeleton file with the original filename** as specified for proper grading.

❖ Remove all printf statements added for debugging before submitting.

❖ There are a total of 6 questions. Please refer to the different point values for each question and solve them accordingly.

❖ You should NOT share your code with other students. Any student found to have a high level of code similarity, as determined by our similarity detection process, may face severe penalties.

1. numOf1s (8 points)

❖ Filename to be submitted : prob1.c

Write a function numOf1s() that takes an integer n as input and counts the number of '1's in its binary representation.

Input

- A single integer ‘n’ where 1 <= n <= 10^9.

Output

- int num that represents the number of ‘1’s in n’s binary representation.

Examples

```
[Test with prob1.c]
./prob1 3 5 7
—-------------------------------—-------------------------------—-----------------------------
[Output]
2 2 3
```

2. Calculator (8 points)

❖ Filename to be submitted : prob2.c

You are required to implement a simple calculator function named ‘calculator’ that performs operations on float numbers.

Input

- The function receives three parameters
    1) char op : A character representing the operator to be used
    2) float a : The first operand (-50.0 <= a <= 50.0)
    3) float b : The second operand (-50.0 <= b <= 50.0)

Output

- The function should perform the following operations based on the ‘op’ parameter.
    - If op is '+', print the sum of a and b.
    - If op is '-', print the result of subtracting b from a.
    - If op is 'x', print the product of a and b.
    - If op is '/', print the result of dividing a by b.
    - If the given operator is not one of the specified operators (+, -, x, /) or if the calculation cannot be performed, print "Invalid"
- The calculation result should be printed using a printf statement within the calculate function.
- Use %.1f to represent values rounded to two decimal places.
- The output should not contain any spaces.

Examples

[Test with prob2.c]

```
./prob2 + 5.0 3.0
```

——————————————————————————————————————

[Output]

```
8.0
```

[Test with prob2.c]

```
./prob2 ^ 5.0 3.0
```

——————————————————————————————————————

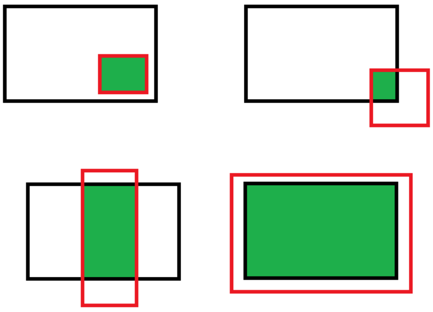
[Output]

```
Invalid
```

3. Overlapped Area of Two Rectangles (12points)

❖ Filename to be submitted : prob3.c

You are provided with a partial implementation of a C program that calculates the overlapping area of two rectangles on a 2D plane. Your task is to complete the implementation of the function `OverlappedArea` that computes this area.



The function `OverlappedArea` should calculate the area where two rectangles overlap. The rectangles are defined by two points:

- p1 (top-left corner)
- p2 (bottom-right corner)

Input

- The function receives two rectangles, `r1` and `r2`, as input parameters.

Output

- The function should return the overlapping area between the two rectangles as a `float`.
- Use `%.2f` to represent values rounded to two decimal places.
- If the rectangles do not overlap, the function should return `0.00`.

Examples

```
[Test with prob3.c]
./prob3.c 1.0 4.0 5.0 1.0 2.0 3.0 6.0 0.0
r1.p1 = (1.0, 4.0), r1.p2 = (5.0, 1.0)
r2.p1 = (2.0, 3.0), r2.p2 = (6.0, 0.0)
—-------------------------------------—-------------------------------------—-------------------------------
>> The overlapped area of the two rectangles is: 6.00
```

4. Fill in the structure (12 points)

❖ Filename to be submitted : prob4.c

The provided ‘main’ function is incomplete. Analyze the ‘main’ function to determine what components are needed and how they should operate. After that, complete the following tasks.

1) Implement the ‘Student’ structure
   - Define the ‘Student’ structure with the members based on the information you can obtain from the ‘main’ function.

2) Implement the ‘set_total’ function
   - The ‘set_total’ function should set the ‘total’ member of a ‘student’ structure to the sum of the ‘middle’ and ‘final’ members.

You **should not modify the ‘main’ function**. All the information needed to implement the ‘Student’ structure and ‘set_total’ function can be obtained from the ‘main’ function.

Examples

[Test with prob4.c]

```
./prob4 20 30 40 10 25 15 45 45 10 10
—-----------------------------—------------------------------—----------------------------
id: 1, total score : 50
id: 2, total score : 50
id: 3, total score : 40
id: 4, total score : 90
id: 5, total score : 20
```

5. Reverse a Linked List (15 points)

❖ Filename to be submitted : prob5.c

You are provided with a partial implementation of a C program that manages a single linked list. Your task is to complete the implementation of the function ‘reverse_list’ that reverses the order of nodes in the linked list.

Input

- The function takes a pointer to the head of the linked list as its parameter

  >> reverse_list(`Node** head`).

Output

- The function does not return any value. It modifies the linked list in place.
- After the function executes, the head pointer should point to the new head of the reversed linked list.

Examples

[Test with prob5.c]

```
./prob5 1 2 3 4 5

>> 5 4 3 2 1
——-------------------------------——-------------------------------——----------------------------
./prob5 4 2 5 7 8 9

>> 9 8 7 5 2 4
```

6. Make a unique string (15 points)

❖ Filename to be submitted : prob6.c

You need to implement a function uniqueCharacters() that reads text from a file and prints a string consisting of unique characters found in the file, excluding spaces. The characters should appear in the order they first appear in the file.

Input

- filename : a filename of the txt file to be processed

Output

- print a string that contains the unique characters (excluding spaces) in the order they first appear in the file.

Examples

```
[ input.txt ]

hello world
—---------------------------------—--------------------------------—-----------------------------
[ Test with prob6.c ]
./prob6 input.txt
—---------------------------------—--------------------------------—-----------------------------
helowrd
```

Thank you :)