Omer Cihan Benzer, 2017400048, Assignment 1 Report

Description:

This code creates definite amounts of cocktail glasses, depending on the given two integers "glassSize" and "strawPos".

glassSize determines the size of the glass.

strawPos determines the position of the straw which also limits the amount of cocktail glasses there should be untill the drink gets low enough.

Solution:

There are 3 main parts in the Assignment: CocktailTop, CocktailMiddle and CocktailBottom. However, CocktailMiddle is not an independent method.

CocktailTop creates the straw untill it comes to the glass.

CocktailMiddle creates the side parts and the drink/straw inside.

CocktailBottom creates the bottom of the glass and the holding place.

There are also 3 different integers to use for loops:

i, increasing every frame

j, meaning the column

k, meaning the line

Implementation:

In CocktailTop, there is a single "o" followed by j times " ", j keeps increasing untill it hits strawPos.

CocktailBottom is made of 2 parts:

The first part is made of glassSize times " " and a single "--" to resemble the bottom of the cocktail glass.

Second part loops glassSize times, which also prints glassSize times " " continuing with "||" .

CocktailMiddle is the complicated part:

Firstly, it prints a single "\" after printing " " for j times {different j from the CocktailTop's j}, j {which is 1 at start} increases every line untill strawPos to create the nice V shape.

Note that to print \, you have to use \\ since \ can mean different things.

After the \, there is an if, relies on the int i from the very beginning:

Every time a glass gets printed, the drink inside of the glass gets lower and lower. i starts as 0 , increasing each frame. with i and j, code writes the following:

—If j is bigger than i, it prints "*" for (2*glassSize +1 -j) times

it is 2*glassSize because both start and end increases if glassSize increases

-j is because the cup gets smaller each line

—If i is bigger than j, it prints " " just like the "*" (2*glassSize +1 -j); however, it also prints a single "o" to continue the straw when it reaches the required position.

This happens when strawPos is equal to (k -j +1). As said, j meaning the column and k meaning the line.

Last of all, the code prints a single / and ends the line.

Output of the program:

I am using (3.4) and (4.2) for the examples (Down below):

Conclusion:

The code does what was required but might've been better. There are some parts where the same for loop is used but was written multiple times.

As example, the for loops in 32th and 37th are quite similar. As another example, 2 parts where straw is written might've been a single part too {lines 4 and 37}.

Another part where it can be improved is the for limits. (k<=2*size-j+1) in the 32th row does the job but looks kind of cheap. It is possible to make it look better for sure, but requires some time since every for loop has to change if i,j,k are changed.

int size and int pos also look a bit weird, it might be possible to rename them to make it look better.

Apart from them, it looks a solid coding as far as I can see.