ITCS 113 – Fundamentals of Programming
Lecture 3: Extra Exercises

No Need to Submit but Practices (ALL) is encouraged.

Q1: Determine the value of in `total` and its counter (e.g. `i)` after each following loops is executed and the number of iterations (rounds) in .

Loops	# of iterations	Value of *counter* and *total*

`int total=10;`
`for(int i=0;i<5;i++)`
`{`
`    total+=i;`
`}`
	5	Loop #1: i=0, total = 10
		Loop #2: i=1, total = 11
		Loop #3: i=2, total = 13
		Loop #4: i=3, total = 16
		Loop #5: i=4, total = 20

`int total = 1;`
`for(int c=0;c<8;c=c+2)`
`{`
`    total *= 2;`
`}`		

`float total = 10.0;`
`for(int j=10;j>=6;j--)`
`{`
`    total=total/2;`
`}`		

`float total = 10.0;`
`int counter=-1;`
`while(counter<3){`
`    total+=counter;`
`    counter+=4;`
`}`		

`int total = 4;`
`int max=16;`
`while(total<=max){`
`    total*=2;`
`}`		

`char total = 'A';`
`char min='C';`
`while(total<=(min+1)){`
`    total++;`
`}`		

Q2: Write the program to show the sequence of “Seven-Up” game between 1 to `n`.
The “Seven-up” game has the following rules:

- The sequence starts from 1 to `n`,
- If a number in the sequence is divisible by 7 or ending with 7, display "Seven-Up" instead of a number.

The program must receive an interger, n and display the “Seven-Up” sequence accordingly.

Sample inputs and output

#	Input	Expected Screen Output
1	30	30 1 2 3 4 5 6 seven-up 8 9 10 11 12 13 seven-up 15 16 seven-up 18 19 20 seven- up 22 23 24 25 26 seven-up seven-up 29 30

Q3: Write the C program to display a multiplication table of an integer `x` and the number of rows, `y`. The program takes two integer inputs: `x` and `y`. If `x` or `y` is negative number or zero, the program display “Unable to create a table”; otherwise, the program shows the multiplication table of `x`, from 1 to `y`.

Requirement: The program must use `while` loop

Sample inputs and output

#	Input	Output	Expected Screen Output
1	3 0	Unable to create a table	3 0 Unable to create a table
2	5 8	5*1=5 5*2=10 5*3=15 5*4=20 5*5=25 5*6=30 5*7=35 5*8=40	5 8 5*1=5 5*2=10 5*3=15 5*4=20 5*5=25 5*6=30 5*7=35 5*8=40
3	-2 5	-2*1=-2 -2*2=-4 -2*3=-6 -2*4=-8 -2*5=-10	-2 5 -2*1=-2 -2*2=-4 -2*3=-6 -2*4=-8 -2*5=-10

Q4: Write a C program to count the number of integers in the given range, `[x, y]`, that can be divisible by `d`. The program takes three inputs: `x`, `y`, and `d:`

- `x` is the start number of the range (inclusive)
- `y` is the ending number of the range (inclusive)
- `d` is the divider

You can assume that all three inputs are always integers greater than 0 and `x<y`.

Requirement: The program must use `for` loop

Sample inputs and output

#	Input	Output	Expected Screen Output

1	`5 21 3`	`6 9 12 15 18 21 count=6`	`5 21 3`
			`6 9 12 15 18 21`
			`count=6`

2	`-10 10 4`	`-8 -4 0 4 8 count=5`	`-10 10 4`
			`-8 -4 0 4 8 count=5`

Q5: Write the program to calculate the average score of a students given multiple homework. The program must repetitively take the input scores until receiving `-1.` Then, the program will calculate the average of the input scores, display the average of the scores, and determine "Pass" and "Fail" of the student.

- You can assume the user will input only numbers and `-1`
- The valid score is in range of 0 and 100 inclusively. If the user input the invalid score > 100, the program will skip the loop and take a new input.
- If the user input `-1`, the program will stop taking inputs and calculate the average of "valid" score.
- If the average score is more than 50, the program displays "Pass"; otherwise, the program displays "Fail"
- All output numbers are in 2 decimal places

Requirement: The program must use `while` loop

Sample inputs and output

#	Input	Output	Expected Screen Output

1	`75`
	`80`
	`43`
	`-5`
	`90`
	`55`
	`q`
		`68.60 Pass`	`75`
			`80`
			`43`
			`-5`
			`90`
			`55`
			`q`
			`68.6 Pass`

2	`35`
	`70`
		`32.50 Fail`	`35`
			`70`

#	Input	Output	Expected Screen Output
	101 25 0 q		101 25 0 q 32.50 Fail
3	90 100 -11 200 q	95.00 Pass	90 100 -11 200 q 95.00 Pass

Q6: Write the program to count how many “Vowel” and “Consonant” user have inputs until they input ‘0’. The program must repetitively take the input alphabets until receiving `'0'.` Then, the program count how many vowels (`'a', 'e', 'i', 'o'` and `'u'`) and consonants (e.g., `'b'`, `'c'`, `'d'`, …) and display the outputs.

- You can assume the user will input only the lowercase alphabets and `'0'`
- If users inputs `'0'`, the program stops taking inputs and displays the output.

Requirement: The program must use loop

Sample inputs and output

#	Input	Output	Expected Screen Output
1	a b c 0	vowel:1 consonant:2	a b c 0 vowel:1 consonant:2
2	i i u 0	vowel:3 consonant:0	i i u 0 vowel:3 consonant:0
3	0	vowel:0 consonant:0	0 vowel:0 consonant:0
4	j y n x 0	j y n x 0 vowel:0 consonant:4	j y n x 0 vowel:0 consonant:4