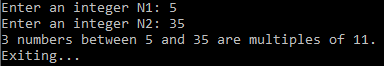
**Lab 2**

**/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////**Q1 Write a program that does the following:

* Asks the user to enter an integer N1.
* Asks the user to enter an integer N2.
* Prints out the number of multiples of 11 that are greater than or equal to M and less than or equal to N.
* Produces an output EXACTLY as shown below:

**/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////**

**output**



**/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////Q2** The extraterrestrials living in the planet Numerion revere a specific type of integers N. In particular, for those creatures, an integer N is holy if there exists an integer i >= 1 such that:

N = i\*i + i

Here are some examples of holy numbers:

2 (because 2 = 1\*1 + 1)

6 (because 6 = 2\*2 + 2)

12 (because 12 = 3\*3 + 3

write a program that:

* Asks the user to enter an integer N.
* Prints out whether that number is a holy number in planet Numerion.
* Prints output EXACTLY as shown below:

**/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////**

**//output**







**/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////**

**Q3** Write a program that:

* Asks the user to enter a positive integer N > 1. It is OK if your program crashes when the user does not enter a valid integer.
* If N <= 1, the program terminates.
* If N > 1, the program prints out N0, N1, N2, ..., up to (and including) the last power of N that is less than or equal to 40000.

**/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////**

**//output**

For example: if the user enters 40, your program output should look EXACTLY like this:





**///////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////// Q4** Write a program satisfying these specs:

* It asks the user to enter an integer called "low".
* It asks the user to enter an integer called "high".
* If low ≤ high, the program prints, in ascending order, all integers between low and high (including low and high), **except for** multiples of 4.
* If low > high, the program just prints "no numbers found" and exits.

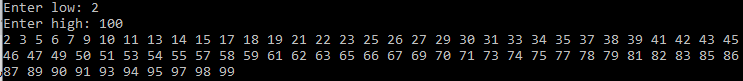
**/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////**

**//output**

For example, if the user enters 7 for low and 13 for high, your program output should look EXACTLY like this:







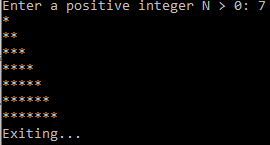
**/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////Q5** Write a program that:

* Asks the user to enter a positive integer N > 0. It is OK if your program crashes when the user does not enter a valid integer.
* If N <= 0, the program terminates.
* If N > 0, the program prints out:
  + A line with one character '\*'.
  + A line with two characters '\*'.
  + A line with three characters '\*'.
  + And so on, up to and including a line with N characters '\*'.

**/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////**

**//output**

For example: if the user enters 7, your program output should look EXACTLY like this:

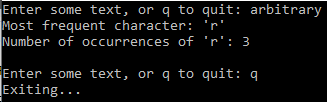
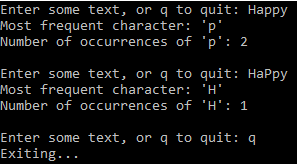


**/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////**

**Q6** Complete that program, by defining a countOccurrences function, and by modifying the existing mostFrequentCharacter function, to satisfy the following specs:

* Function countOccurrences takes two arguments, a string, and a character. It returns the number of times the character occurs in the string.
* Function mostFrequentCharacter takes a string as an argument. It returns the character that occurs the most times in the string. If multiple characters tie for occurring the most times, the function can return any one of those characters tying for most occurrences. Any of those return values will be considered correct.
* You are not allowed to change the main program in any way.

**///////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////output**



**#include <stdio.h>**

**#include <string.h>**

**#include <stdbool.h>**

**....... countOccurrences(..........) {**

**........**

**}**

**...... mostFrequentCharacter(.......) {**

**}**

**int main() {**

**char text[21];**

**while (true) {**

**printf("Enter some text, or q to quit: ");**

**gets(text);**

**if (strcmp(text,"q")==0) {**

**break;**

**}**

**if (strlen(text) == 0) {**

**break;**

**}**

**char c = mostFrequentCharacter(text);**

**int number = countOccurrences(text, c);**

**printf("Most frequent character: '%c'\n", c);**

**printf("Number of occurrences of '%c': %d\n\n", c, number);**

**}**

**printf("Exiting...\n");**

**return 0;**

**}**

**/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////**

**Q7** Complete the program so that it:

* Asks the user to enter a word.
* Prints out the number of occurrences of the letter B (both upper and lower case) in that word.
* You are not allowed to alter the main function in any way.

**#include <stdio.h>**

**#include <string.h>**

**#include <stdbool.h>**

**...... countBs(.....)**

**{**

**}**

**int main() {**

**char text[21];**

**printf("Please enter a word: ");**

**gets(text);**

**int result = countBs(text);**

**printf("%s contains letter B %d times.\n", text, result);**

**return 0;**

**}**

**///////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////output**



**/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////Q8** Complete the program so that :

* Function **lowered()** returns a lower case version of the string passed on to it.
* Function reversed returns a reversed version of the string passed on to it.
* Produces output as shown below.
* You are not allowed to alter the main function in any way.

**#include <stdio.h>**

**....... lowered(.....) {**

**}**

**....... reversed(.......) {**

**}**

**int main(void) {**

**char str1[21]="helLLLLOOO";**

**char str2[21]="helLooo";**

**char str3[21];**

**printf("Original %s\n", str1);**

**printf("lowered %s\n", lowered(str1));**

**strcpy(str2,lowered(str1));**

**strcpy(str3,reversed(str2));**

**printf("reversed %s\n", str3);**

**}**

**///////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////output**



**/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////Q9** Complete that program, by doing the following: pickMiddle takes three arguments, called first, second, third.

* Complete the function pickMiddle() It has three parameters (a, b, c) and returns the middle of the values of the three arguments.
* Function user\_integer has one parameter, called message, which is a string.
* Function user\_integer  prints out the message, accepts a string and converts it to an integer using **atoi.**
* Produces output as shown below.

**#include <stdio.h>**

**#include <string.h>**

**.... user\_integer(....)**

**{**

**........**

**}**

**..... pickMiddle( a, b, c){**

**........**

**}**

**int main(void) {**

**int N1 = user\_integer("Enter number N1: ");**

**int N2 = user\_integer("Enter number N2: ");**

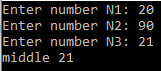
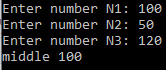
**int N3 = user\_integer("Enter number N3: ");**

**printf("middle %d\n", pickMiddle(N1,N2,N3));**

**return 0;**

**}**

**///////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////output**



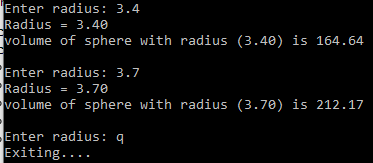
**/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////Q10** Complete that program, by defining a volume function, and by modifying the existing user\_message function, so as to satisfy the following specs:

* Produces output as shown below.
* Function user\_message displays the message and accepts a string; if the string is “q”, allows the user to exits- otherwise converts the string to a double and returns it.
* Function volume takes one argument, called radius.
* Function volume returns the volume of a sphere with the specified radius, using the formula:

volume = (4/3) \* π \* radius3

**/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////**

**//output**



#include <stdio.h>

**.....**

**.... user\_msg(.....)**

**{**

**......**

**}**

**......volume(...){**

**.......**

**}**

**int main(void) {**

**double R=0;**

**while (true){**

**R = user\_msg("\nEnter radius: ");**

**printf("Radius = %0.2f\n",R);**

**printf("volume of sphere with radius (%0.2f) is %0.2f\n",R, volume(R));**

**}**

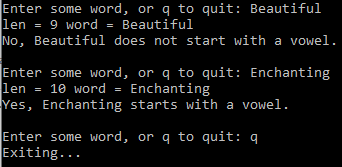
**return 0;**

**}**

**/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////Q11** Complete that program, by defining a startsWithVowel function, that satisfies the following specs:

* Function startsWithVowel takes one argument, called word.
* if word starts with a vowel (i.e., an a, e, i, o, or u, in upper or lower case), then startsWithVowel(word) returns true.
* otherwise, startsWithVowel(word) returns false.
* Produces output as follows:

**///////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////output**



**#include <stdio.h>**

**#include <string.h>**

**#include <stdbool.h>**

**.......startsWithVowel(.....)**

**{**

**..................**

**}**

**void main(void)**

**{**

**char word[21];**

**while (true)**

**{**

**printf("Enter some word, or q to quit: ");**

**gets(word);**

**if (strcmp(word,"q")==0)**

**{**

**printf("Exiting...\n");**

**exit(0);**

**}**

**bool result = startsWithVowel(word);**

**if (result == true)**

**{**

**printf("Yes, %s starts with a vowel.\n\n", word);**

**}**

**else**

**{**

printf("No, %s does not start with a vowel.\n\n", word);

}

}

}

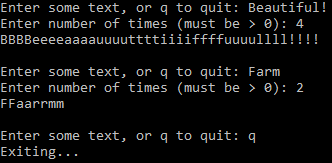
**/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////**

Q12 Complete that program, by defining a repeatLetters function, that satisfies the following specs:

* Function **repeatLetters()** takes two arguments, called text, times.
* The function goes through the letters of text in the order in which they appear in text, and prints each such letter as many times as specified by the argument times.
* Function **userInteger()** accepts a message, displays it and accepts a string. If the user enters “q”, exit – otherwise convert the string to an integer and return the value.
* Produces output as shown below.
* You are not allowed to modify the main function in any form.

**/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////**

**//output**



**#include <stdio.h>**

**........ include files as needed**

**...... userInteger(........)**

**{.......................**

**}**

**....... repeatLetters(.......)**

**{**

**...........**

**}**

**void main()**

**{**

**while (true)**

**{**

**printf("Enter some text, or q to quit: ");**

**char text[200];**

**gets(text);**

**if (strcmp(text,"q")==0)**

**{**

**printf("Exiting...\n");**

**exit(0);**

**}**

**int times = userInteger("Enter number of times (must be > 0): ");**

**repeatLetters(text, times);**

**printf(" \n\n");**

**}**

**}**

**/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////**