COMPILER DESIGN LAB BCSE307P
LAB 1

Name: Pragya Sekar
Regno: 23BCE1491

Question 1

Aim: To write a c program to accept any number of a's followed by any number of b's end with one a.

Algorithm:

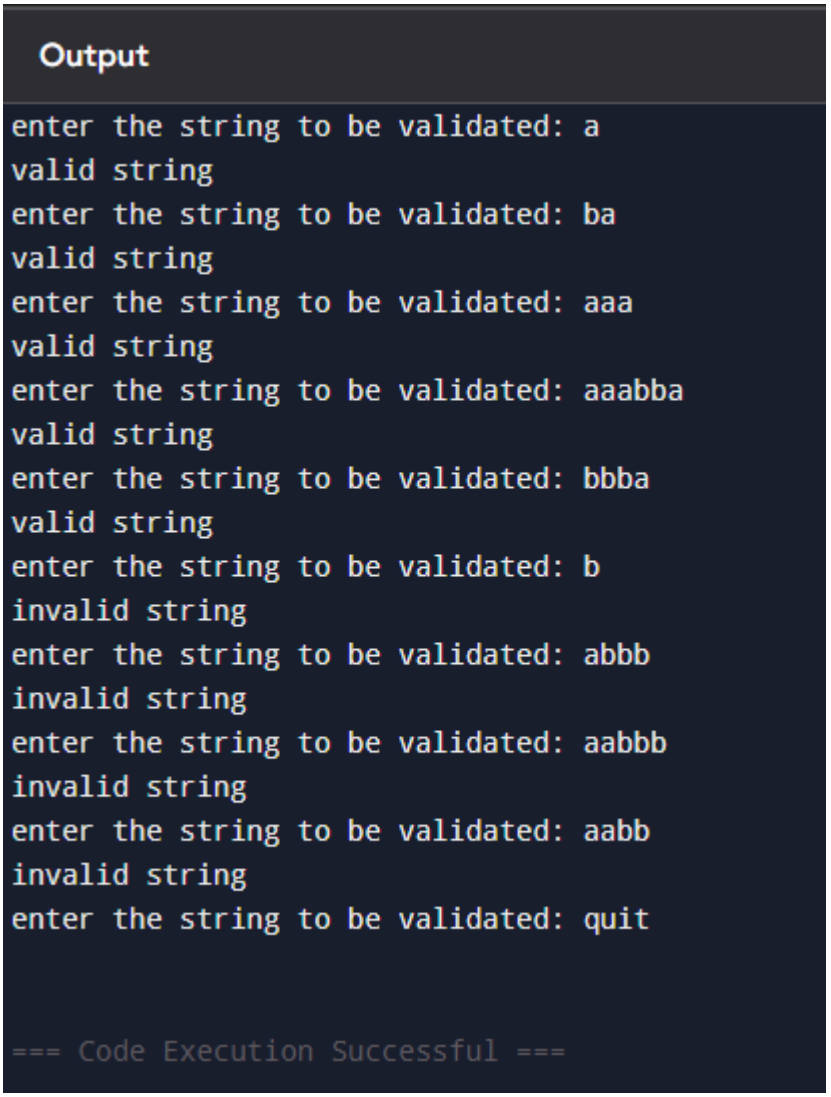
1. Check is the string is empty, if it is, then return false
2. Scan through the a's and keep count
3. Scan through the b's and keep count
4. The last letter of the string must be a, then return true
5. Else return false

Code:

```
#include <string.h>
#include <stdbool.h>

bool question1(char* text){
    int n = strlen(text);
    if(n==0 || text[n-1]!='a') return false;

    int i = 0;
    while(i<n && text[i]=='a') i++;
    while(i<n-1 && text[i]=='b') i++;
    if(text[n-1]=='a') return true;

    return false;

}

int main() {
    char text[100];
    printf("enter the string to be validated: ");
    scanf("%s",text);

    while(strcmp(text,"quit")!=0){
        if(question1(text)){
            printf("valid string\n");
        }else{
            printf("invalid string\n");
        }
         printf("enter the string to be validated: ");
```

```
        scanf("%s",text);
    }

}
```

Sample input and output:



Results: Successfully demonstrated with inputs as above and verified the output.

- Accepted: "a", "ba", "aaa", "aaabba", "bbba"
- Rejected: "b", "abbb", "aabbb", "aabb"

Question 2

Aim: To write a C program that checks if a binary string contains an even number of zeros.

Algorithm:

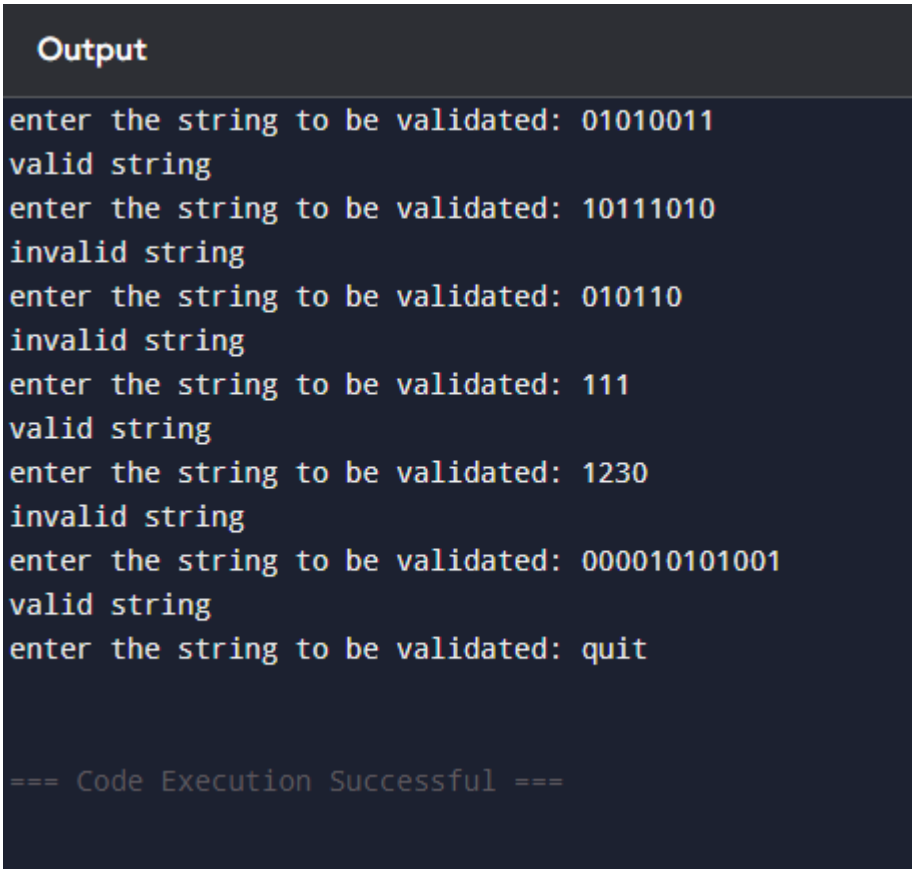
1. Initialise counter variable to keep track of number of zeros
2. Iterate through each character in the given text
    A. Upon a occurrence of zero, increment the counter
3. Check if the counter is even or odd
    A. If even, return true
    B. Else return false

Code:

```
#include <stdio.h>
#include <string.h>
#include <stdbool.h>

bool question2(char* text){
    int n = strlen(text);
    int c = 0;

    for(int i=0;i<n;i++){
        if(text[i]=='0'){
            c++;
        }else if(text[i]!='1'){
            return false;
        }
    }

    return (c%2==0);
}

int main() {
    char text[100];
    printf("enter the string to be validated: ");
    scanf("%s",text);

    while(strcmp(text,"quit")!=0){
        if(question2(text)){
            printf("valid string\n");
        }else{
            printf("invalid string\n");
        }
         printf("enter the string to be validated: ");
         scanf("%s",text);
    }
```

}

Sample input and output:



Results: Successfully demonstrated with inputs as above and verified the output.

- Accepted: “01010011”, “111”, “000010101001”
- Rejected: “10111010”, “010110”, “1230”

Question 3

Aim: To Write a C program to check the mobile number pattern and validate it

- Must be exactly 10 digits
- Must start with 6, 7, 8, or 9
- Should contain only digits

Algorithm:

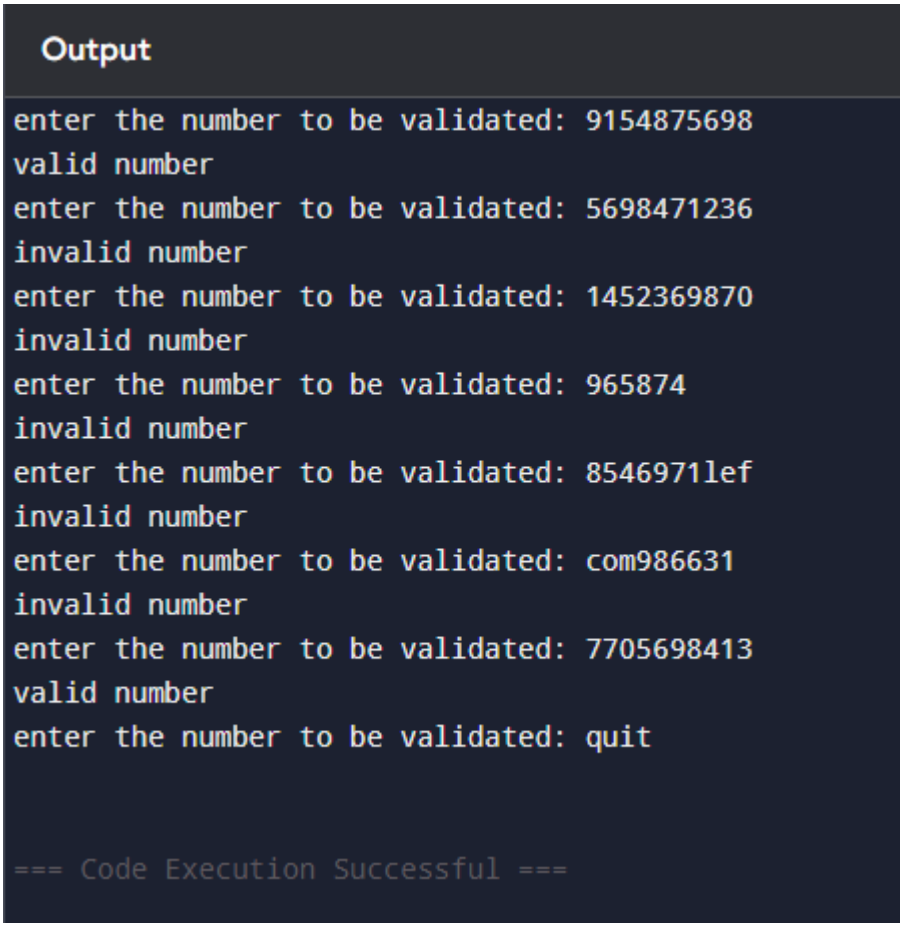
1. Check if the number's length is 10, if not return false
2. Check is the first digit of the number is 6,7,8,9, is not return false
3. Iterate through each character in the number and check if it is a digit, if not return false
4. If all the conditions satisfy, return true

Code:

```
#include <stdio.h>
#include <string.h>
#include <stdbool.h>
#include <ctype.h>

bool question3(char* text){
    int n = strlen(text);
    if(n!=10) return false;
    if(text[0]!='9'&& text[0]!='8'&& text[0]!='7'&& text[0]!='6') return false;

    for(int i=0;i<n;i++){
        if(!isdigit(text[i])) return false;
    }

    return true;
}

int main() {
    char text[100];
    printf("enter the number to be validated: ");
    scanf("%s",text);

    while(strcmp(text,"quit")!=0){
        if(question3(text)){
            printf("valid number\n");
        }else{
            printf("invalid number\n");
        }
         printf("enter the number to be validated: ");
         scanf("%s",text);
    }
```

}

Sample input and output:



Results: Successfully demonstrated with inputs as above and verified the output.

- Accepted: “9154875698”, “7705698413”
- Rejected: “5698471236”, “1452369870”, “965874”, “8546971lef”, “com986631”