ELP 718

Telecommunication Software Lab

INDIAN INSTITUTE OF TECHNOLOGY
NEW DELHI

Assignment 6

September 12, 2017

Name: **GHANENDRA SINGH**

Entry Number: **2017JTM2022**

Contents

1 Problem Statement 1

Parity Check The simplest way of error detection is to append a single bit , called a parity check, to a string of data bits. This parity check bit has the value 1 if number of 1's in the bit string is even and has the value 0 otherwise, i.e., Odd Parity Check.

Bit Oriented Framing Data Link Layer needs to pack bits into frames, so that each frame is distinguishable from another. Frames can be fixed or variable size. In variable size framing, we define end of frame using bit oriented approach. It uses a special string of bits, called a flag for both idle fill and to indicate the beginning and the ending of frames. The string 0101 is used as the bit string or flag to indicate the end of the frame. The bit stuffing rule is to insert a 0 after each appearance of 010 in the original data. In addition, if the frame ends in 01, a 0 would be stuffed after the 1st 0 in the actual terminating string 0101.

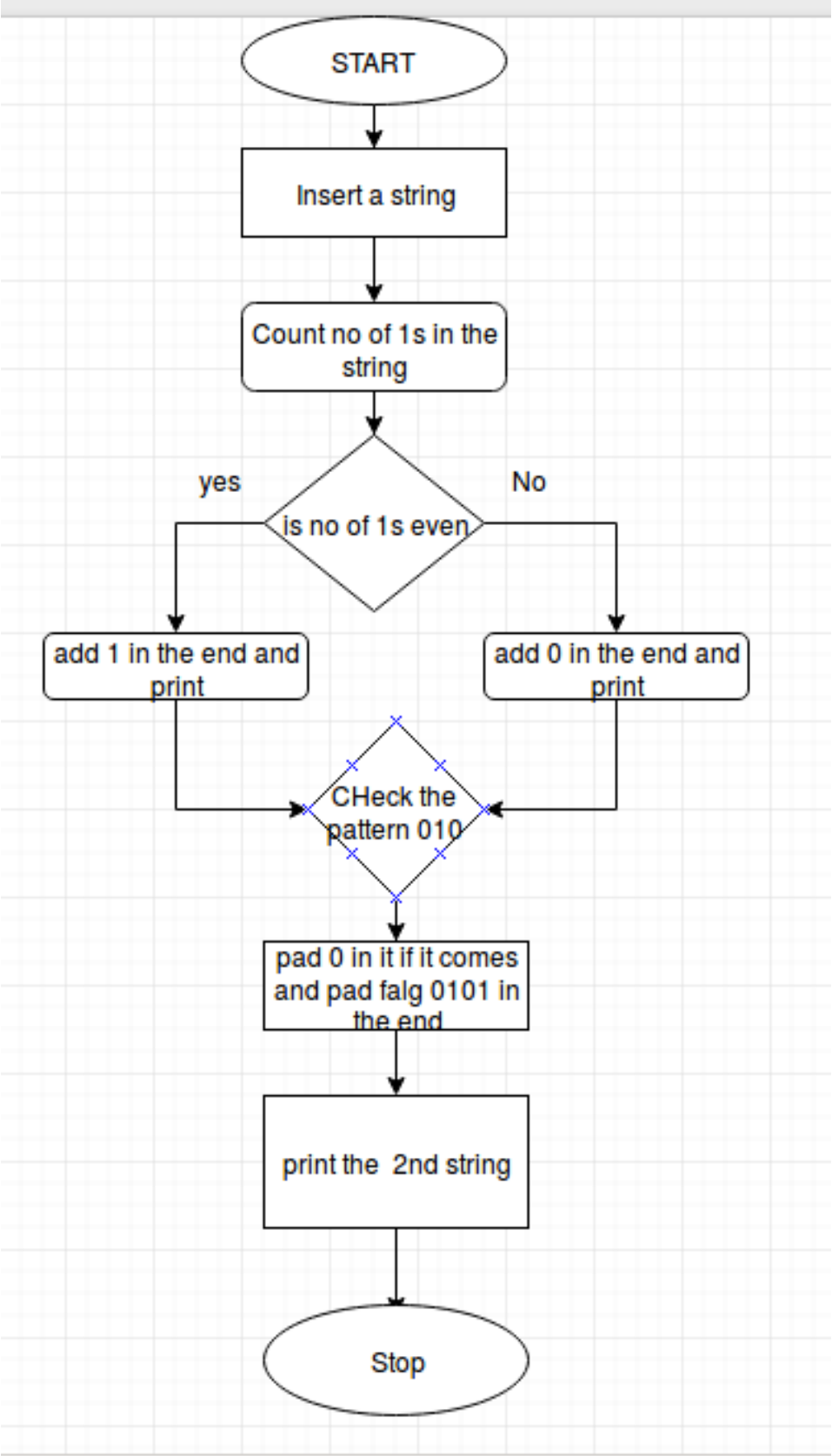
1.1 Assumptions

The string is inserted for bit stufing

1.2 Program Structure

1. fisrt insert a string and count no of 1s
2. Add 0 and 1 according yo parity
3. show 1 string
4. add flag and pad 0 to he string
5. show the second string

1.3 Algorithm



1.4 Implementation

1.5 Input and Output Format

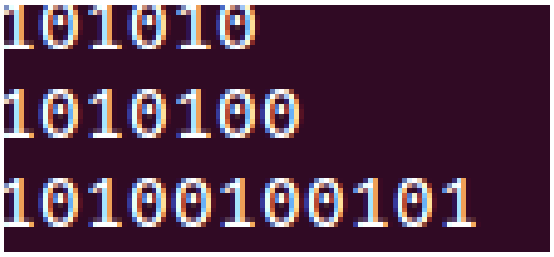
Input format
Enter binary bit data which has to be transmitted.
01010

Output format
Print binary bit data with parity bit.
Print the modified string received at the other end.
010101
0100100100101

1.6 Test Cases

Input
101010
Output
1010100
10100100101

1.7 Screenshots



2 Problem Statement 2

•3X3 Numeric Tic-Tac-Toe (Use numbers 1 to 9 instead of X's and O's) One player plays with the odd numbers (1, 3, 5, 7, 9) and other player plays with the even numbers (2,4,6,8). All numbers can be used only once. The player who puts down 15 points in a line wins (sum of 3 numbers). Always Player with odd numbers start the game. Once a line contains two numbers whose sum is 15 or greater, there is no way to complete that line, although filling in the remaining cell might be necessary to complete a different line. Note – Line can be horizontal, vertical or diagonal

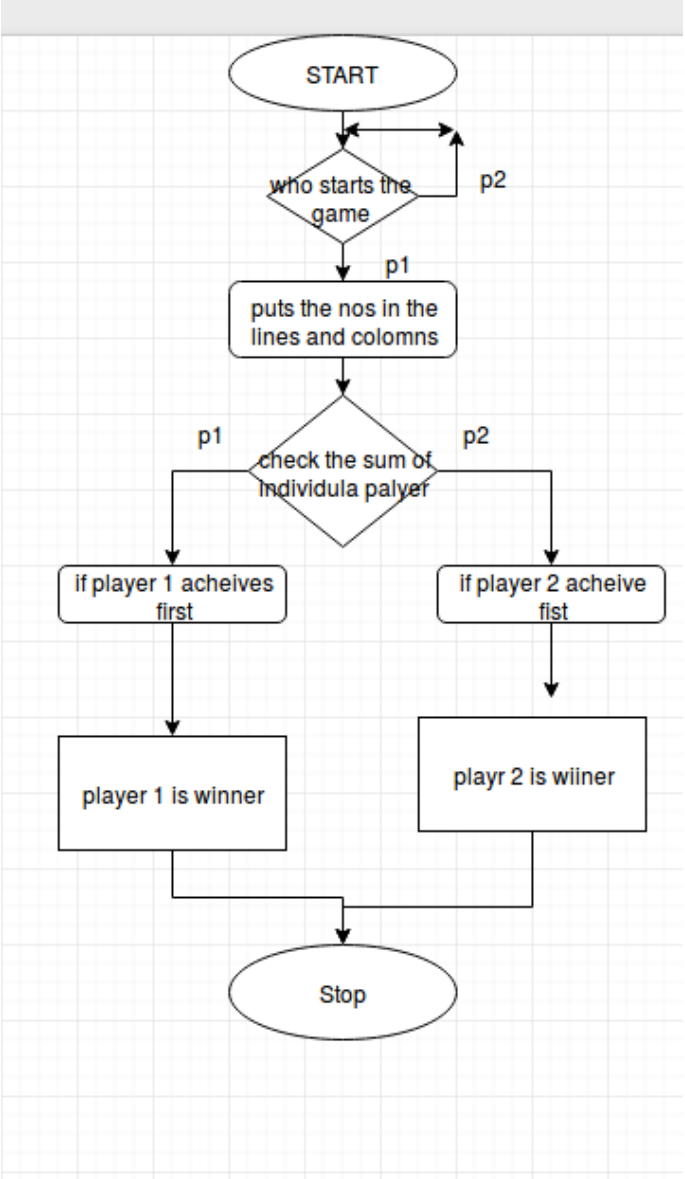
2.1 Assumptions

The psoition can be only between 1-9 The number can be only between 1-9 The sum can be only between 1-15

2.2 Program Structure

1. first take the input from the 1 person
2. take the inputs from the sceond person
3. check the sum of indicidul player in a line
4. who got first sum of 15 will win

2.3 Algorithm



2.4 Implementation

2.5 Input and Output Format

Constraints:
1¡=Position¡=9
1¡=Number¡=9
1¡=Sum¡=15

Terminal:
Print 'Welcome to the Game!'.
Print whether it is Player 1's or Player 2's chance.
Get the position and number to be entered from user.
Show tic tac toe with data.
Continue till the game gets draw or some player wins and show result.
Ask user whether to continue for next game or exit.

2.6 Test Cases

2.7 Screenshots

3 Bibliography

1. https://stackoverflow.com/questions/1155617/count-occurrence-of-a-character-in-a-string
2. https://docs.python.org/3/
3. https://git-scm.com/download/linux
4. https://git-scm.com/download/linux

4 Code

code1

```
###### this is the first .py file ###########

####### write your code here ##########

#parity checking

N=input()                #taking the input string and conveting into binary
N=str(N)

x= N.count('1')              #counting the no of ones in string
if (x % 2) == 0:             #loop for inserting 1 and 0 in the sttring
    print(N+ '1')
else:
    print(N+'0')                #printing accordingly


#bit stuffing
j=0
while (j<len(N)):
    a=N.find('010')                     #finding the location of 010 on the fram
    m='0'
    a=a+2                             #increaisng he index in order to get the outpu
    n=N[:a]+'0'+N[a:]                   #padding 0 at the end of 010
    j=j+1



print(n+'0101')          #joining flag in the end
```

code2

```
###### this is the second .py file ###########

####### write your code here ##########

def tic_tac_toe():
    board = [1, 2, 3, 4, 5, 6, 7, 8, 9]
    end = False
    win_commbinations = ((0, 1, 2), (3, 4, 5), (6, 7, 8), (0, 3, 6), (1, 4, 7),

    def draw():
        print(board[0], board[1], board[2])
        print(board[3], board[4], board[5])
        print(board[6], board[7], board[8])
        print()

    def p1():
        n = choose_number()
        if board[n] == "X" or board[n] == "O":
            print("\nYou can't go there. Try again")
            p1()
        else:
            board[n] = "X"

    def p2():
        n = choose_number()
        if board[n] == "X" or board[n] == "O":
            print("\nYou can't go there. Try again")
            p2()
        else:
            board[n] = "O"

    def choose_number():
        while True:
            while True:
                a = input()
                try:
                    a  = int(a)
                    a -= 1
                    if a in range(0, 9):
                        return a
                    else:
                        print("\nThat's not on the board. Try again")
                        continue
                except ValueError:
                    print("\nThat's not a number. Try again")
                    continue
```

```
    def check_board():
        count = 0
        for a in win_commbinations:
            if board[a[0]] == board[a[1]] == board[a[2]] == "X":
                print("Player 1 Wins!\n")
                print("Congratulations!\n")
                return True

            if board[a[0]] == board[a[1]] == board[a[2]] == "O":
                print("Player 2 Wins!\n")
                print("Congratulations!\n")
                return True
        for a in range(9):
            if board[a] == "X" or board[a] == "O":
                count += 1
            if count == 9:
                print("The game ends in a Tie\n")
                return True

    while not end:
        draw()
        end = check_board()
        if end == True:
            break
        print("Player 1 choose where to place a cross")
        p1()
        print()
        draw()
        end = check_board()
        if end == True:
            break
        print("Player 2 choose where to place a nought")
        p2()
        print()

    if input("Play again (y/n)\n") == "y":
        print()
        tic_tac_toe()

tic_tac_toe()
```