# Introduction to Simple Error Detection in Data Communication

# (Odd/ Even Parity checker)

## Introduction:

Develop a client-server application; implement odd parity checking mechanism on the messages. The programs should deliver the following:

**Client:**

**Step 01:** The client should read from an input file (input.txt), one line at a time.

**Step 02:**Split the line in to characters.

**Step 03:** Convert each character to corresponding ASCII value.

**Step 04:** Calculate the odd parity bit of each character.

**Step 05:**Construct each character such as:

Binary representation of **A**: 01000001

Odd parity Bit: 1

Modified representation should be: 1000001**1**

*Bonus: Introduce random error in the parity bit.*

**Step 06:** Send the modified representation of each character to the server.

**Server:**

**Step 01:** Sever will receive the characters from the client.

**Step 02:** Verify the odd parity bit calculation of each character.

* If correct, write the actual character sent by the client in a file (output.txt)
* Otherwise, display an error message.

## Example:

**Client Side**

For each character

Send to the **server**

**Convert ’A’ to its ASCII:**

**value**= 65

**Shift the binary representation 1 bit to the left.**

For value:

01000001 after shifting 10000010

**Calculate the number of ones in the binary representation**

For **value** its 2 (even)

**Split the String in to Character array:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A |  | f | o | r |  | A | p | p | l | e |

**Read the first Line:**

String line=”A for Apple”

**Input.txt**

A for Apple

B for Ball

C for Cat

If # of 1’s is odd,

value | 0

Else, value | 1

e.g. 10000010| 1= 10000011

**Server Side:**

odd

Write the corresponding character to output.txt

Value = right shift value for 1 bit

even

Display error message

Receive (**value**)

Check if # of 1’s in **value** is odd or not?