**PRACTICAL – 03**

**STOP AND WAIT PROTOCOL PROGRAM**

#include<iostream>

#include <time.h>

#include <cstdlib>

#include<ctime>

#include <unistd.h>

using namespace std;

class timer {

private:

unsigned long begTime;

public:

void start() {

begTime = clock();

}

unsigned long elapsedTime() {

return ((unsigned long) clock() - begTime) / CLOCKS\_PER\_SEC;

}

bool isTimeout(unsigned long seconds) {

return seconds >= elapsedTime();

}

};

int main()

{

int frames[] = {1,2,3,4,5,6,7,8,9,10};

unsigned long seconds = 5;

srand(time(NULL));

timer t;

cout<<"Sender has to send frames : ";

for(int i=0;i<10;i++)

cout<<frames[i]<<" ";

cout<<endl;

int count = 0;

bool delay = false;

cout<<endl<<"Sender\t\t\t\t\tReceiver"<<endl;

do

{

bool timeout = false;

cout<<"Sending Frame : "<<frames[count];

cout.flush();

cout<<"\t\t";

t.start();

if(rand()%2)

{

int to = 24600 + rand()%(64000 - 24600) + 1;

for(int i=0;i<64000;i++)

for(int j=0;j<to;j++) {}

}

if(t.elapsedTime() <= seconds)

{

cout<<"Received Frame : "<<frames[count]<<" ";

if(delay)

{

cout<<"Duplicate";

delay = false;

}

cout<<endl;

count++;

}

else

{

cout<<"---"<<endl;

cout<<"Timeout"<<endl;

timeout = true;

}

t.start();

if(rand()%2 || !timeout)

{

int to = 24600 + rand()%(64000 - 24600) + 1;

for(int i=0;i<64000;i++)

for(int j=0;j<to;j++) {}

if(t.elapsedTime() > seconds )

{

cout<<"Delayed Ack"<<endl;

count--;

delay = true;

}

else if(!timeout)

cout<<"Acknowledgement : "<<frames[count]-1<<endl;

}

}while(count!=10);

return 0;

}

**OUTPUT :**

Sender has to send frames : 1 2 3 4 5 6 7 8 9 10

Sender Receiver

Sending Frame : 1 Received Frame : 1

Acknowledgement : 1

Sending Frame : 2 ---

Timeout

Sending Frame : 2 ---

Timeout

Sending Frame : 2 Received Frame : 2

Delayed Ack

Sending Frame : 2 Received Frame : 2 Duplicate

Delayed Ack

Sending Frame : 2 Received Frame : 2 Duplicate

Delayed Ack

Sending Frame : 2 Received Frame : 2 Duplicate

Delayed Ack

Sending Frame : 2 Received Frame : 2 Duplicate

Delayed Ack

Sending Frame : 2 ---

Timeout

Sending Frame : 2 Received Frame : 2 Duplicate

Acknowledgement : 2

Sending Frame : 3 Received Frame : 3

Delayed Ack

Sending Frame : 3 ---

Timeout

Sending Frame : 3 Received Frame : 3 Duplicate

Acknowledgement : 3

Sending Frame : 4 Received Frame : 4

Delayed Ack

Sending Frame : 4 Received Frame : 4 Duplicate

Delayed Ack

Sending Frame : 4 Received Frame : 4 Duplicate

Delayed Ack

Sending Frame : 4 Received Frame : 4 Duplicate

Acknowledgement : 4

Sending Frame : 5 ---

Timeout

Sending Frame : 5 Received Frame : 5

Delayed Ack

Sending Frame : 5 Received Frame : 5 Duplicate

Delayed Ack

Sending Frame : 5 Received Frame : 5 Duplicate

Delayed Ack

Sending Frame : 5 Received Frame : 5 Duplicate

Delayed Ack

Sending Frame : 5 Received Frame : 5 Duplicate

Delayed Ack

Sending Frame : 5