Practical File

COMPUTER NETWORKING

Paper Code: CEC013

Bachelor of Engineering

(2016-2020)

COE

Submitted by:

Yash Bansal

2016UCO1639

COE-3

Semester IV

INDEX

1. **Implementation of data link layer framing method such as character stuffing and bit stuffing.**

BIT STUFFING :

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

int main(){
        int a[20],b[20],i,j,k,count,n;
        printf("Enter frame size : ");
        scanf("%d",&n);
        printf("Enter the frame in the form of 0s & 1s : ");
        for(i=0;i<n;i++){
                scanf("%d",&a[i]);
        }
        i=0;
        count=1;
        j=0;
        while(i<n){
                b[j]=a[i];
                if(a[i]==1){
                        for(k=i+1;a[k]==1 && count<5 && k<n;k++){
                                j++;
                                b[j]=a[k];
                                count++;
                                if(count==5){
```

```
                                        j++;
                                        b[j]=0;
                                    }
                                    i=k;
                                }
                            }
                            i++;
                            j++;
                        }

                        printf("Afer bit stuffing : ");
                        for(i=0;i<j;i++){
                            printf("%d",b[i]);
                        }
                        printf("\n")
                        return 0;
                }
```

OUTPUT:

```
Sals-MacBook-Pro:Lab yashbansal$ gcc Bitstuffing.c
Sals-MacBook-Pro:Lab yashbansal$ ./a.out
Enter frame size : 8
Enter the frame in the form of 0s & 1s : 0 1 1 1 1 1 1 0
Afer bit stuffing : 011111010
Sals-MacBook-Pro:Lab yashbansal$
```

CHARACTER STUFFING:

```
#include <iostream>
#include <cstring>
using namespace std;

string charstuff(string s,char start,char end);
string decodecharstuff(string s,char start,char end);

int main(){
    string str;
    cout<<"Enter Data: ";
    cin>>str;
    char start;
    cout<<"Enter Starting delimiter: ";
    cin>>start;
    char end;
    cout<<"Enter Ending delimiter: ";
    cin>>end;
    string result = charstuff(str, start, end);
    cout<<"Stuffed Data: "<<result<<endl;
    cout<<"De-Stuffed Data: "<<decodecharstuff(result, start, end)<<endl;
}

string charstuff(string s,char start,char end){
    string res;
    res += start;
```

```
    for(int i=0; i<s.length(); i++) {
        if(s[i] == start) {
            res += s[i];
            res += s[i];
        }

        else if(s[i]==end) {
            res += s[i];
            res += s[i];
        }
        else {
            res+=s[i];
        }
    }
            res+=end;
    return res;
}

string decodecharstuff(string s,char start,char end){
    string data;
    for(int i=1; i<s.length()-1; i++) {
        if(s[i]==start && i!=s.length()-2) {
            if(s[i+1]==start) {
                data += s[i];
                i++;
            }
```

```
        }
        else if(s[i]==end && i!=s.length()-2) {
            if(s[i+1]==end) {
                data += s[i];
                i++;
            }
        }
        else {
            data += s[i];
        }
            }
    return data;
}
```

OUTPUT:

```
Sals-MacBook-Pro:Lab yashbansal$ g++ charStuffing.cpp
Sals-MacBook-Pro:Lab yashbansal$ ./a.out
Enter Data: YashBansal
Enter Starting delimiter: *
Enter Ending delimiter: *
Stuffed Data: *YashBansal*
De-Stuffed Data: YashBansal
Sals-MacBook-Pro:Lab yashbansal$
```

2. Implementation of CRC (Cyclic Redundancy Check) error detection method.

```
#include <iostream>
using namespace std;

int main(){
    int disze, dasze, divisor[100], data[100], rem[100], quo[100], temp[100];
    cout<<"Enter the size of divisor : ";
    cin>>disze;
    cout<<"Input divisor : ";
    for (int i = 0; i < disze; ++i){
        cin>>divisor[i];
    }
    cout<<"Enter the size of data : ";
    cin>>dasze;
    cout<<"Input data : ";
    for (int i = 0; i < dasze; ++i){
        cin>>data[i];
    }
    for (int i = 0; i < disze -1; ++i){
        data[dasze+i] = 0;
    }
    int j = 0, k = 0;
    for(int i = 0; i < disze; ++i){
        temp[i] = data[i] ^ divisor[i];
        ++k;
    }
```

```
quo[j++] = 1;
while(k != dasze + disze - 1){
        for(int i = 0; i<disze - 1; ++i){
                temp[i] = temp[i+1];
        }
        temp[disze - 1] = data[k++];
        int ctr = 0;
        while(temp[ctr] != 1){
                for(int i = 0; i<disze - 1; ++i){
                        temp[i] = temp[i+1];
                }
                temp[disze - 1] = data[k++];
                quo[j++] = 0;
        }
        for (int i = 0; i < disze; ++i){
                temp[i] = temp[i] ^ divisor[i];
        }
        quo[j++]=1;
}
for(int i = 0; i < disze - 1; i++){
        data[dasze + i] = temp[i+1];
}
cout<<"Transmitted data is : "<<endl;
for (int i = 0; i < dasze + disze - 1 ; ++i){
        cout<<data[i];
}
```

```
        cout<<endl;

        return 0;

}
```

OUTPUT:

```
[Sals-MacBook-Pro:Lab yashbansal$ g++ crc.cpp
[Sals-MacBook-Pro:Lab yashbansal$ ./a.out
 Enter the size of divisor : 5
 Input divisor : 1 1 0 0 1
 Enter the size of data : 8
 Input data : 1 0 1 0 1 1 0 1
 Transmitted data is :
 101011011001
 Sals-MacBook-Pro:Lab yashbansal$
```

3. **Implementation of hamming code (7, 4) to limit the noise**

```
#include <iostream>

using namespace std;


int main() {

   int data[10];

   int dataatrec[10],c,c1,c2,c3,i;


   cout<<"Enter 4 bits of data one by one:\n";

   cin>>data[0];

   cin>>data[1];

   cin>>data[2];
```

```
cin>>data[4];

//Calculation of even parity
data[6]=data[0]^data[2]^data[4];
data[5]=data[0]^data[1]^data[4];
data[3]=data[0]^data[1]^data[2];

cout<<"\nEncoded data is : ";
for(i=0; i<7; i++)
   cout<<data[i];

cout<<"\n\nEnter received data bits one by one : ";
for(i=0; i<7; i++)
   cin>>dataatrec[i];

c1 = dataatrec[6]^dataatrec[4]^dataatrec[2]^dataatrec[0];
c2 = dataatrec[5]^dataatrec[4]^dataatrec[1]^dataatrec[0];
c3 = dataatrec[3]^dataatrec[2]^dataatrec[1]^dataatrec[0];
c = c3*4 + c2*2 + c1 ;

if(c==0) {
   cout<<"\nNo error while transmission of data.\n";
}

else {
   cout<<"\nError at position "<<c;
```

```
        cout<<"\nData sent : ";
        for(i=0; i<7; i++)
            cout<<data[i];

        cout<<"\nData received : ";
        for(i=0; i<7; i++)
            cout<<dataatrec[i];

        cout<<"\nCorrect message is : ";

        //if errorneous bit is 0 we complement it, else vice versa
        if(dataatrec[7-c]==0)
            dataatrec[7-c]=1;
        else
            dataatrec[7-c]=0;
        for (i=0; i<7; i++) {
            cout<<dataatrec[i];
        }
    }
    return 0;
}
```

OUTPUT:

```
[Sals-MacBook-Pro:Lab yashbansal$ g++ hamming.cpp
[Sals-MacBook-Pro:Lab yashbansal$ ./a.out
 Enter 4 bits of data one by one:
 1 0 1 0

 Encoded data is : 1010010

 Enter received data bits one by one : 1 0 1 0 0 1 0

 No error while transmission of data.
 Sals-MacBook-Pro:Lab yashbansal$
```

4. **Implementation of LZW Compression Algorithm**

```
#include <string>

#include <map>

#include <iostream>

#include <iterator>

#include <vector>


 using namespace std;


// Compress a string to a list of output symbols.

// The result will be written to the output iterator

// starting at "result"; the final iterator is returned.

template <typename Iterator>

Iterator compress(const string &uncompressed, Iterator result){

 // Build the dictionary.

        int dictSize = 256;
```

```
        map<string,int> dictionary;
        for (int i = 0; i < 256; i++)
                dictionary[string(1, i)] = i;

        string w;
        for (string::const_iterator it = uncompressed.begin();
                it != uncompressed.end(); ++it) {
                        char c = *it;
                        string wc = w + c;
                        if (dictionary.count(wc))
                        w = wc;
                        else {
                                *result++ = dictionary[w];
                                // Add wc to the dictionary.
                                dictionary[wc] = dictSize++;
                                w = string(1, c);
                        }
        }

        // Output the code for w.
        if (!w.empty())
                *result++ = dictionary[w];
        return result;
}

// Decompress a list of output ks to a string.
```

```
// "begin" and "end" must form a valid range of ints
template <typename Iterator>
string decompress(Iterator begin, Iterator end){
	// Build the dictionary.
	int dictSize = 256;
	map<int,string> dictionary;
	for (int i = 0; i < 256; i++)
		dictionary[i] = string(1, i);

	string w(1, *begin++);
	string result = w;
	string entry;
	for ( ; begin != end; begin++) {
		int k = *begin;
		if (dictionary.count(k))
			entry = dictionary[k];
		else if (k == dictSize)
			entry = w + w[0];
		else
			throw "Bad compressed k";

		result += entry;

		// Add w+entry[0] to the dictionary.
		 dictionary[dictSize++] = w + entry[0];
```

```
            w = entry;
        }
        return result;
    }

    int main(){
        vector<int> compressed;
        string str;
        cout<<"Enter the string to be compressed: ";
        cin>>str;
        cout<<"String after compression is--> ";
        compress(str, back_inserter(compressed));
        copy(compressed.begin(), compressed.end(), ostream_iterator<int>(cout, ",
    "));
        cout << endl;
        cout<<"String after Decompression is--> ";
        string decompressed = decompress(compressed.begin(), compressed.end());
        cout << decompressed << endl;
        return 0;
    }
```

OUTPUT:

```
[Sals-MacBook-Pro:Lab yashbansal$ g++ lzw.cpp                                   ]
[Sals-MacBook-Pro:Lab yashbansal$ ./a.out                                       ]
Enter the string to be compressed: YashBansal
String after compression is--> 89,      97,     115,    104,    66,     97,    1
10,     115,    97,     108,
String after Decompression is--> YashBansal
Sals-MacBook-Pro:Lab yashbansal$
```

5. Program to find shortest path routing between nodes (Dijkstra's Algorithm)

```
import java.util.*;


public class djikstra {
        static final int V = 9 ;
        int min_distance(int dist[] , boolean sptSet[]) {
                int min = Integer.MAX_VALUE;
                int min_index = -1;
                for ( int v = 0 ; v < V ; v++ ) {
                        if (sptSet[v]==false && dist[v]<=min) {
                                min = dist[v] ;
                                min_index = v ;
                        }
                }
                return min_index;
        }

        void print_sol(int dist[],int n) {
                System.out.println("Vertex   Distance from Source");
                for (int i = 0; i < V; i++)
                        System.out.println("  "+i+"            "+dist[i]);
        }


        void dijk(int graph[][] , int src) {
                int dist[] = new int [V];
```

```
        boolean sptSet[] = new boolean [V];
        for (int i = 0; i < V; i++) {
      dist[i] = Integer.MAX_VALUE;
      sptSet[i] = false;
    }

        dist[src] = 0 ;

        for ( int count = 0 ; count < V-1 ; count++) {
                int u = min_distance(dist,sptSet);
                sptSet[u] = true ;

                for (int v = 0 ; v < V ; v++) {
                        if (!sptSet[v] && graph[u][v]!=0 && dist[u] !=
                                Integer.MAX_VALUE && dist[u]+graph[u][v]
                                < dist[v]){
                                dist[v] = dist[u] + graph[u][v];
                        }
                }
        }

        print_sol(dist,V);
  }

public static void main (String[] args) {
   int graph[][] = new int[][]{{0, 4, 0, 0, 0, 0, 0, 8, 0},
```

```
                        {4, 0, 8, 0, 0, 0, 0, 11, 0},
                        {0, 8, 0, 7, 0, 4, 0, 0, 2},
                        {0, 0, 7, 0, 9, 14, 0, 0, 0},
                        {0, 0, 0, 9, 0, 10, 0, 0, 0},
                        {0, 0, 4, 14, 10, 0, 2, 0, 0},
                        {0, 0, 0, 0, 0, 2, 0, 1, 6},
                        {8, 11, 0, 0, 0, 0, 1, 0, 7},
                        {0, 0, 2, 0, 0, 0, 6, 7, 0}
                        };
    djikstra t = new djikstra();
    t.dijk(graph, 0);
          }
}
```

OUTPUT:

```
[Sals-MacBook-Pro:Lab yashbansal$ javac djikstra.java
[Sals-MacBook-Pro:Lab yashbansal$ java djikstra
 Vertex   Distance from Source
   0              0
   1              4
   2              12
   3              19
   4              21
   5              11
   6              9
   7              8
   8              14
 Sals-MacBook-Pro:Lab yashbansal$
```

6. Implementation of RSA Algorithm.

```
#include <iostream>
using namespace std;

int gcd(int a,int b){
   if(a < b)
      swap(a,b);
   if(b == 0)
      return a;
   return gcd(b,a%b);
}

int modularExponentiation(int a, int b, int c){
   int ans = 1;
   while(b != 0) {
      if((b & 1) == 1)
         ans = (ans%c * a%c)%c;
      a = (a*a)%c;
      b = b/2;
   }
   return ans;
}

int main(){
   string a = "0ABCDEFGHIJKLMNOPQRSTUVWXYZ";
```

```
int p,q;
cout<<"Enter two prime numbers : "<<endl;
cin>>p>>q;
int n = p*q;

while(n<26){
    cout<<"Value of p and q is not large enough."<<endl;
    cout<<"Please enter p and q such that p*q is greater than 26 : "<<endl;
    cin>>p>>q;
    n = p*q;
}

int phi = (p-1)*(q-1);
int e,d;

for(int i=2; i<phi; i++){
    if(gcd(i,phi) == 1){
        e = i;
        break;
    }
}

for(int i=1; i<phi; i++){
    if((e*i)%phi == 1){
        d = i;
        break;
```

```
    }
}

cout<<"n: "<<n<<endl;
cout<<"phi: "<<phi<<endl;
cout<<"e: "<<e<<endl;
cout<<"d: "<<d<<endl;
string p1;

cout<<"Enter the plain text : "<<endl;
cin>>p1;
cout<<endl;
cout<<"Encryption"<<endl;
int s;
int* c = new int[p1.length()];

for(int i=0; i<p1.length(); i++){
    for(int j=1; j<a.length(); j++){
        if(a[j] == p1[i]){
            s = j;
            break;
        }
    }
    c[i] = modularExponentiation(s,e,n);
    cout<<s<<endl;
    cout<<"cipher: "<<c[i]<<endl;
```

```
    }

    cout<<endl;
    cout<<"Decryption"<<endl;

    for(int i=0; i<p1.length(); i++) {
        int m1 = modularExponentiation(c[i],d,n);
        cout<<a[m1];
    }

    cout<<endl;
    return 0;
}
```

OUTPUT :

```
Sals-MacBook-Pro:Lab yashbansal$ g++ rsa.cpp
Sals-MacBook-Pro:Lab yashbansal$ ./a.out
Enter two prime numbers :
7 11
n: 77
phi: 60
e: 7
d: 43
Enter the plain text :
YASHBANSAL

Encryption
25
cipher: 53
1
cipher: 1
19
cipher: 68
8
cipher: 57
2
cipher: 51
1
cipher: 1
14
cipher: 42
19
cipher: 68
1
cipher: 1
12
cipher: 12

Decryption
YASHBANSAL
Sals-MacBook-Pro:Lab yashbansal$
```

7. Write a socket program to implement a listener and a talker.

Server :

// Server side C/C++ program to demonstrate Socket programming

```
#include <stdio.h>
#include <sys/socket.h>
#include <stdlib.h>
#include <netinet/in.h>
#include <string.h>
#define PORT 8080
intmain(intargc, charconst*argv[])
{
   intserver_fd, new_socket, valread;
   structsockaddr_in address;
   intopt = 1;
   intaddrlen = sizeof(address);
   charbuffer[1024] = {0};
   char*hello = "Hello from server";

   // Creating socket file descriptor
   if((server_fd = socket(AF_INET, SOCK_STREAM, 0)) == 0)
   {
      perror("socket failed");
      exit(EXIT_FAILURE);
   }

   // Forcefully attaching socket to the port 8080
   if(setsockopt(server_fd, SOL_SOCKET, SO_REUSEADDR | SO_REUSEPORT,
                                          &opt, sizeof(opt)))
   {
      perror("setsockopt");
      exit(EXIT_FAILURE);
   }
   address.sin_family = AF_INET;
   address.sin_addr.s_addr = INADDR_ANY;
   address.sin_port = htons( PORT );

   // Forcefully attaching socket to the port 8080
   if(bind(server_fd, (structsockaddr *)&address,
                           sizeof(address))<0)
   {
      perror("bind failed");
      exit(EXIT_FAILURE);
```

```
    }
    if(listen(server_fd, 3) < 0)
    {
        perror("listen");
        exit(EXIT_FAILURE);
    }
    if((new_socket = accept(server_fd, (structsockaddr *)&address,
                    (socklen_t*)&addrlen))<0)
    {
        perror("accept");
        exit(EXIT_FAILURE);
    }
    valread = read( new_socket , buffer, 1024);
    printf("%s\n",buffer );
    send(new_socket , hello , strlen(hello) , 0 );
    printf("Hello message sent\n");
    return 0;
}
```

Client :

```
// Client side C/C++ program to demonstrate Socket programming

#include <stdio.h>
#include <sys/socket.h>
#include <stdlib.h>
#include <netinet/in.h>
#include <string.h>
#define PORT 8080

intmain(intargc, charconst*argv[])
{
    structsockaddr_in address;
    intsock = 0, valread;
    structsockaddr_in serv_addr;
    char*hello = "Hello from client";
    charbuffer[1024] = {0};
    if((sock = socket(AF_INET, SOCK_STREAM, 0)) < 0)
    {
        printf("\n Socket creation error \n");
        return-1;
    }

    memset(&serv_addr, '0', sizeof(serv_addr));

    serv_addr.sin_family = AF_INET;
```

```
    serv_addr.sin_port = htons(PORT);

    // Convert IPv4 and IPv6 addresses from text to binary form
    if(inet_pton(AF_INET, "127.0.0.1", &serv_addr.sin_addr)<=0)
    {
       printf("\nInvalid address/ Address not supported \n");
       return-1;
    }

    if(connect(sock, (structsockaddr *)&serv_addr, sizeof(serv_addr)) < 0)
    {
       printf("\nConnection Failed \n");
       return-1;
    }
    send(sock , hello , strlen(hello) , 0 );
    printf("Hello message sent\n");
    valread = read( sock , buffer, 1024);
    printf("%s\n",buffer );
    return0;
}
```

Server :

```
pc@pc -Lenovo-G50-80:~$ gcc server.c -o server
server.c: In function ‘main’:
server.c:53:15: warning: implicit declaration of function ‘read’; did you
mean ‘fread’? [-Wimplicit-function-declaration]
valread = read( new_socket , buffer, 1024);
^~~~
fread
pc@pc -Lenovo-G50-80:~$ ./server
Hello from client
Hello message sent
```

Client :

```
pc@pc-Lenovo-G50-80:~$ gcc client.c -o client
client.c: In function ‘main’:
client.c:28:8: warning: implicit declaration of function ‘inet_pton’ [-
Wimplicit-function-declaration]
if(inet_pton(AF_INET, "127.0.0.1", &serv_addr.sin_addr)<=0)
^~~~~~~~
client.c:41:15: warning: implicit declaration of function ‘read’; did you
mean ‘fread’? [-Wimplicit-function-declaration]
valread = read( sock , buffer, 1024);
^~~~
fread
pc@pc -Lenovo-G50-80:~$ ./client
Hello message sent
Hello from server
```