Experiment – 1 Stream Ciphers

1.1) Write a C Program to implement Shift Cipher.

Program:

```
#include<stdio.h>
#include<ctype.h>
int main() {
char text[500], ch; int key;
printf("Enter a message to decrypt: ");
scanf("%s", text);
printf("Enter the key: ");
scanf("%d", & key);
for (int i = 0; text[i] != '\0'; ++i) {
ch = text[i];
if (isalnum(ch)) {
if (islower(ch)) {
ch = (ch - 'a' - key + 26) % 26 + 'a';
}
if (isupper(ch)) {
ch = (ch - 'A' - key + 26) % 26 + 'A';
}
if (isdigit(ch)) {
ch = (ch - '0' - key + 10) % 10 + '0';
}}
else {printf("Invalid Message");
}
text[i] = ch;
}
printf("Decrypted message: %s", text);
return 0;
}
```

Output:

```
C:\Users\admin\Desktop\EXP1.exe
Enter a message to decrypt: siri
Enter the key: 4
Decrypted message: oene
--------------------------------
Process exited after 9.729 seconds with return value 0
Press any key to continue . . .
```

Result: Thus the implementation of Caesar cipher had been executed successfully

1.2) Write a C Program to implement Mono-Alphabetic Substitution Cipher

Program:

```
#include<stdio.h>
char monocipher_encr(char);
char alpha[27][3]= {{'a','f'},{'b','a'},{'c','g'},{'d','u'},{'e','n'},{'f','i'},{'g','j'},{'h','k' },{'i','l'},{'j',
'm'},{'k','o'},{'l','p'},{'m','q'},{'n','r'},{'o','s'},{'p','t'},{'q','v'},{'r','w'},{'s','x'},{'t','y'},'v','b'},{{'u'
,'z'},{'w','c'},{'x','d'},{'y','e'},{'z','h'}};
char str[20];
int main() {
char str[20], str2[20];
int i;
printf("\n Enter String:");
gets(str);
for (i = 0; str[i]; i++) {
str2[i] = monocipher_encr(str[i]);
}
str2[i] = '\0';
printf("\n Before Decryption:%s", str);
printf("\n After Decryption:%s\n", str2);
}
char monocipher_encr(char a) {
int i;
for (i = 0; i< 27; i++) {
if (a == alpha[i][0])
break;
}
return alpha[i][1];
}
```

Output:

```
C:\Users\admin\Desktop\EXP2.exe

 Enter String:aditya

 Before Decryption:aditya
 After Decryption:fulyef

--------------------------------
Process exited after 2.061 seconds with return value 0
Press any key to continue . . .
```

Experiment – 2 Block Ciphers

2.1) Write a C Program to implement one-time pad cipher.

Program:

```
#include<stdio.h>
#include<string.h>
#include<ctype.h>
main(){
int i,j,len1,len2,numstr[100],numkey[100],numcipher[100];
char str[100],key[100],cipher[100];
printf("Enter a string text to encrypt\n");
gets(str);
for(i=0,j=0;i<strlen(str);i++){
if(str[i]!=' '){
str[j]=toupper(str[i]);
j++;
}}
str[j]='\0';
for(i=0;i<strlen(str);i++){
numstr[i]=str[i]-'A';
}
printf("Enter key string of random text\n");
gets(key);
for(i=0,j=0;i<strlen(key);i++){
if(key[i]!=' '){
key[j]=toupper(key[i]);
j++;
}}
key[j]='\0';
for(i=0;i<strlen(key);i++)
numkey[i]=key[i]-'A';
for(i=0;i<strlen(str);i++)
numcipher[i]=numstr[i]+numkey[i];
for(i=0;i<strlen(str);i++){
if(numcipher[i]>25){
numcipher[i]=numcipher[i]-26;
}}
printf("One Time Pad Cipher text is\n");
for(i=0;i<strlen(str);i++){
printf("%c",(numcipher[i]+'A'));
}
printf("\n");
}
```

Output:

```
C:\Users\admin\Desktop\EXP3.exe
Enter a string text to encrypt
aditya
Enter key string of random text
3
One Time Pad Cipher text is
3G T>A

--------------------------------
Process exited after 6.244 seconds with return value 0
Press any key to continue . . .
```

2.2) Write a C Program to implement vernam cipher.

Program:

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
void encrypt(char *plaintext, char *key,char *ciphertext){
int i;
for(i=0; i<strlen(plaintext);i++){
ciphertext[i] = plaintext[i] ^ key[i];
}}
void decrypt(char *ciphertext , char *key , char *plaintext){
int i;
for(i=0; i<strlen(ciphertext);i++){
plaintext[i] = ciphertext[i] ^ key[i];
}}
int main(int argc, char *argv[]){
char plaintext[100];
char key[100];
char ciphertext[100];
printf("Enter Plaintext ");
scanf("%s",plaintext);
printf("Enter Key ");
scanf("%s",key);
encrypt(plaintext,key,ciphertext);
printf("Ciphertext : %s\n",ciphertext);
decrypt(ciphertext,key,plaintext);
printf("Plaintext: %s\n",plaintext);
return 0;
}
```

Output:

```
C:\Users\admin\Desktop\exp3cns.exe
Enter Plaintext vel
Enter Key bak
Ciphertext : ¶♦
Plaintext: vel     ╤%@

--------------------------------
Process exited after 13.76 seconds with return value 0
Press any key to continue . . .
```

Experiment – 4 Asymmetric Cryptography

4.1) Write a C Program to implement RSA algorithm.

Program:

```
#include<stdio.h>
#include<math.h>
int gcd(int a, int h){
int temp;
while(1){
temp = a%h;
if(temp==0)
return h;
a = h;
h = temp;
}}
int main(){
double p = 3,q = 7;
double n=p*q,count,totient = (p-1)*(q-1);
double e=2;
while(e<totient){
count = gcd(e,totient);
if(count==1)
break;
else e++;
}
double d,k = 2;
d = (1 + (k*totient))/e;
double msg = 12;
double c = pow(msg,e);
double m = pow(c,d);
c=fmod(c,n);
m=fmod(m,n);
printf("Message data = %lf",msg);
printf("\np = %lf",p);
printf("\nq = %lf",q);
printf("\nn = pq = %lf",n);
printf("\ntotient = %lf",totient);
printf("\ne = %lf",e);
printf("\nd = %lf",d);
printf("\nEncrypted data = %lf",c);
printf("\nOriginal Message Sent = %lf",m);
return 0;
}
```

Output:

C:\Users\ADMIN\Desktop\rsa.exe

```
Message data = 12.000000
p = 3.000000
q = 7.000000
n = pq = 21.000000
totient = 12.000000
e = 5.000000
d = 5.000000
Encrypted data = 3.000000
Original Message Sent = 12.000000
--------------------------------
Process exited after 6.628 seconds with return value 0
Press any key to continue . . .
```

4.2) Write a C Program to implement Diffie-Helman Key Exchange Algorithm.

Program:

```
#include<stdio.h>
#include<math.h>
long long int power(long long int a, long long int b, long long int P){
if (b == 1)
return a;
else
return (((long long int)pow(a, b)) % P);
}
int main(){
long long int P, G, x, a, y, b, ka, kb;
P = 23;
printf("The value of P : %lld\n", P);
G = 9;
printf("The value of G : %lld\n\n", G);
a = 4;
printf("The private key a for Alice : %lld\n", a);
x = power(G, a, P);
b = 3;
printf("The private key b for Bob : %lld\n\n", b);
y = power(G, b, P);
ka = power(y, a, P);
kb = power(x, b, P);
printf("Secret key for the Alice is : %lld\n", ka);
printf("Secret Key for the Bob is : %lld\n", kb);
return 0;
}
```

Output:

```
C:\Users\admin\Documents\diffie-hallman.exe
The value of P : 23
The value of G : 9

The private key a for Alice : 4
The private key b for Bob : 3

Secret key for the Alice is : 9
Secret Key for the Bob is : 9

--------------------------------
Process exited after 0.02664 seconds with return value 0
Press any key to continue . . .
```

4.3) Write a C Program to implement Elgamal Cryptographic System.

Program:

```
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <math.h>
int e1, e2,p, d,C1, C2;
FILE *out1, *out2;
int gcd(int a, int b){
int q, r1, r2, r;
if (a > b){
r1 = a;
r2 = b;
}
else {
r1 = b;
r2 = a;
}
while (r2 > 0){
q = r1 / r2;
r = r1 - q * r2;
r1 = r2;
r2 = r;
}
return r1;
}
int FastExponention(int bit, int n, int* y, int* a){
if (bit == 1) {
*y = (*y * (*a)) % n;
}
*a = (*a) * (*a) % n;
}
int FindT(int a, int m, int n){
int r,y = 1;
while (m > 0){
r = m % 2;
FastExponention(r, n, &y, &a);
m = m / 2;
}
return y;
}
int PrimarityTest(int a, int i){
```

```
int n = i – 1, k = 0, m, T;
while (n % 2 == 0){
k++;
n = n / 2;
}
m = n;
T = FindT(a, m, i);
if (T == 1 || T == i - 1) {
return 1;
}
int j;
for (j = 0; j < k; j++){
T = FindT(T, 2, i);
if (T == 1) {
return 0;
}
if (T == i - 1) {
return 1;
}}
return 0;
}
int PrimitiveRoot(int p){
int flag,a;
for (a = 2; a < p; a++){
flag = 1;
int i;
for (i = 1; i < p; i++){
if (FindT(a, i, p) == 1 && i < p - 1) {
flag = 0;
}
else if (flag && FindT(a, i, p) == 1 && i == p - 1) {
return a;
}}}}
int KeyGeneration(){
do {
do
p = rand() + 256;
while (p % 2 == 0);
} while (!PrimarityTest(2, p));
p = 107;
e1 = 2;
do{
d = rand() % (p - 2) + 1;
```

```
} while (gcd(d, p) != 1);
d = 67;
e2 = FindT(e1, d, p);
}
int Encryption(int Plaintext){
out1 = fopen("cipher1.txt", "a+");
out2 = fopen("cipher2.txt", "a+");
int r;
do {
r = rand() % (p - 1) + 1; // 1 < r < p
}
while (gcd(r, p) != 1);
C1 = FindT(e1, r, p);
C2 = FindT(e2, r, p) * Plaintext % p;
fprintf(out1, "%d ", C1);
fprintf(out2, "%d ", C2);
fclose(out1);
fclose(out2);
}
int Decryption(int C1, int C2){
FILE* out = fopen("result.txt", "a+");
int decipher = C2 * FindT(C1, p - 1 - d, p) % p;
fprintf(out, "%c", decipher);
fclose(out);
}
int main(){
FILE *out, *inp;
out = fopen("result.txt", "w+");
fclose(out);
out = fopen("cipher1.txt", "w+");
fclose(out);
out = fopen("cipher2.txt", "w+");
fclose(out);
KeyGeneration();
inp = fopen("plain.txt", "r+");
if (inp == NULL){
printf("Error opening Source File.\n");
exit(1);
}
while (1){
char ch = getc(inp);
if (ch == EOF) {
break; // M < p
```

```
}
Encryption(toascii(ch));
}
fclose(inp);
FILE *inp1, *inp2;
inp1 = fopen("cipher1.txt", "r");
inp2 = fopen("cipher2.txt", "r");
int C1, C2;
while (1){
int ret = fscanf(inp1, "%d", &C1);
fscanf(inp2, "%d", &C2);
if (ret == -1) {
break;
}
Decryption(C1, C2);
}
fclose(inp1);
fclose(inp2);
return 0;
}
```

Output:

```
Enter a prime number: 223
Enter the private key: 23
Enter the generator: 19
Enter the plain text: Elgamal
Enter the sender key: 31

Plain text: Elgamal

Encrypted Message: ◆UO◆ъU

Decrypted Message: Elgamal
```

Experiment – 7 TCP Server Applications

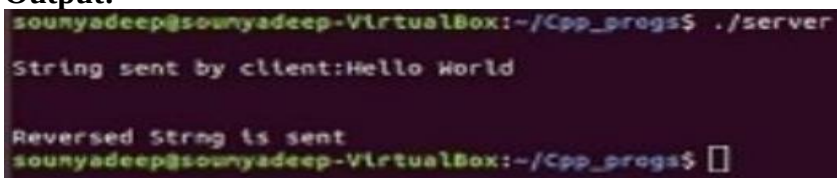
7.1) Design TCP iterative Client and server application to reverse the given input sentence.

Program:

TCP Server:

```
#include<string.h>
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>
#include<sys/socket.h>
#include<sys/types.h>
#define MAXLINE 20
#define SERV_PORT 5777
main(int argc,char *argv) {
int i,j;
ssize_t n;
char line[MAXLINE],revline[MAXLINE];
int listenfd,connfd,clilen;
struct sockaddr_in servaddr,cliaddr;
listenfd=socket(AF_INET,SOCK_STREAM,0);
bzero(&servaddr,sizeof(servaddr));
servaddr.sin_family=AF_INET; servaddr.sin_port=htons(SERV_PORT);
bind(listenfd,(struct sockaddr*)&servaddr,sizeof(servaddr));
listen(listenfd,1);
for( ; ; ) {
clilen=sizeof(cliaddr);
connfd=accept(listenfd,(struct sockaddr*)&cliaddr,&clilen);
printf("connect to client");
while(1) {
if((n=read(connfd,line,MAXLINE))==0)
break;
line[n-1]='\0';
j=0;
for(i=n-2;i>=0;i--)
revline[j++]=line[i];
revline[j]='\0';
write(connfd,revline,n);
}}}
```

Output:



TCP Client:

```
#include<string.h>
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<sys/types.h>
#define MAXLINE 20
#define SERV_PORT 5777
main(int argc,char *argv){
char sendline[MAXLINE],revline[MAXLINE];
int sockfd;
struct sockaddr_in servaddr;
sockfd=socket(AF_INET,SOCK_STREAM,0);
bzero(&servaddr,sizeof(servaddr));
servaddr.sin_family=AF_INET;
servaddr.sin_port=ntohs(SERV_PORT);
connect(sockfd,(struct sockaddr*)&servaddr,sizeof(servaddr));
printf("\n enter the data to be send");
while(fgets(sendline,MAXLINE,stdin)!=NULL){
write(sockfd,sendline,strlen(sendline));
printf("\n line send");
read(sockfd,revline,MAXLINE);
printf("\n reverse of the given sentence is : %s",revline);
printf("\n");
}
exit(0);
}
```

Output:

7.2) Design TCP client and server application to transfer file.

Program:

TCP Server:

```
#include <stdio.h>
#include <netdb.h>
#include <netinet/in.h>
#include <stdlib.h>
#include <string.h>
#include <sys/socket.h>
#include <sys/types.h>
#include <unistd.h> // read(), write(), close()
#define MAX 80
#define PORT 8080
#define SA struct sockaddr
void func(int connfd) // Function designed for chat between client and server.
{
char buff[MAX];
int n;
for (;;) {
bzero(buff, MAX);
read(connfd, buff, sizeof(buff)); // read the message from client and copy it in buffer
printf("From client: %s\t To client : ", buff);
bzero(buff, MAX);
n = 0;
while ((buff[n++] = getchar()) != '\n') // copy server message in the buffer
write(connfd, buff, sizeof(buff)); // and send that buffer to client
if (strncmp("exit", buff, 4) == 0) {
printf("Server Exit...\n");
break;
}}}
int main(){
int sockfd, connfd, len;
struct sockaddr_in servaddr, cli;
sockfd = socket(AF_INET, SOCK_STREAM, 0);
if (sockfd == -1) {
printf("socket creation failed...\n");
exit(0);
}
printf("Socket successfully created..\n");
bzero(&servaddr, sizeof(servaddr));
servaddr.sin_family = AF_INET;
servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
```

```
servaddr.sin_port = htons(PORT);
if ((bind(sockfd, (SA*)&servaddr, sizeof(servaddr))) != 0) {
printf("socket bind failed...\n");
exit(0);
}
else
printf("Socket successfully binded..\n");
if ((listen(sockfd, 5)) != 0) {
printf("Listen failed...\n");
exit(0);
}
else
printf("Server listening..\n");
len = sizeof(cli);
connfd = accept(sockfd, (SA*)&cli, &len);
if (connfd < 0) {
printf("server accept failed...\n");
exit(0);
}
else
printf("server accept the client...\n");
func(connfd);
close(sockfd);
}
```

OUTPUT:

```
Socket successfully created..
Socket successfully binded..
Server listening..
server accept the client...
From client: hi
     To client : hello
From client: exit
     To client : exit
Server Exit...
```

TCP Client:

```
#include <arpa/inet.h>
#include <netdb.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <strings.h>
#include <sys/socket.h>
#include <unistd.h>
#define MAX 80
#define PORT 8080
#define SA struct sockaddr
void func(int sockfd){
char buff[MAX];
int n;
for (;;) {
bzero(buff, sizeof(buff));
printf("Enter the string : ");
n = 0;
while ((buff[n++] = getchar()) != '\n');
write(sockfd, buff, sizeof(buff));
bzero(buff, sizeof(buff));
read(sockfd, buff, sizeof(buff));
printf("From Server : %s", buff);
if ((strncmp(buff, "exit", 4)) == 0) {
printf("Client Exit...\n");
break;
}}}
int main(){
int sockfd, connfd;
struct sockaddr_in servaddr, cli;
sockfd = socket(AF_INET, SOCK_STREAM, 0);
if (sockfd == -1) {
printf("socket creation failed...\n");
exit(0);
}
else
printf("Socket successfully created..\n");
bzero(&servaddr, sizeof(servaddr));
servaddr.sin_family = AF_INET;
servaddr.sin_addr.s_addr = inet_addr("127.0.0.1");
servaddr.sin_port = htons(PORT);
if (connect(sockfd, (SA*)&servaddr, sizeof(servaddr))!= 0) {
```

```
printf("connection with the server failed...\n");
exit(0);
}
else
printf("connected to the server..\n");
func(sockfd);
close(sockfd);
}
```

OUTPUT:

```
Socket successfully created..
connected to the server..
Enter the string : hi
From Server : hello
Enter the string : exit
From Server : exit
Client Exit...
```

Experiment – 10 IPC

Implement the following forms of IPC. a) Pipes b) FIFO

Program: (Pipes)

```
#include<stdio.h>
#include<unistd.h>
int main() {
int pipefds[2];
int returnstatus;
int pid;
char writemessages[2][20]={"Hi", "Hello"};
char readmessage[20];
returnstatus = pipe(pipefds);
if (returnstatus == -1) {
printf("Unable to create pipe\n");
return 1;
}
pid = fork();
if (pid == 0) {
read(pipefds[0], readmessage, sizeof(readmessage));
printf("Child Process - Reading from pipe – Message 1 is %s\n", readmessage);
read(pipefds[0], readmessage, sizeof(readmessage));
printf("Child Process - Reading from pipe – Message 2 is %s\n", readmessage);
} else { //Parent process
printf("Parent Process - Writing to pipe - Message 1 is %s\n", writemessages[0]);
write(pipefds[1], writemessages[0],
sizeof(writemessages[0]));
printf("Parent Process - Writing to pipe - Message 2 is %s\n", writemessages[1]);
write(pipefds[1], writemessages[1], sizeof(writemessages[1]));
}
return 0;
}
```

OUTPUT:

```
Parent Process - Writing to pipe - Message 1 is Hi
Parent Process - Writing to pipe - Message 2 is Hello
Child Process - Reading from pipe – Message 1 is Hi
Child Process - Reading from pipe – Message 2 is Hello
```

Program:(FIFO)

```
#include <stdio.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <fcntl.h>
#include <unistd.h>
#include <string.h>
#define FIFO_FILE "MYFIFO"
int main() {
int fd;
int end_process;
int stringlen;
char readbuf[80];
char end_str[5];
printf("FIFO_CLIENT: Send messages, infinitely, to end enter \"end\"\n");
fd = open(FIFO_FILE, O_CREAT|O_WRONLY);
strcpy(end_str, "end");
while (1) {
printf("Enter string: ");
fgets(readbuf, sizeof(readbuf), stdin);
stringlen = strlen(readbuf);
readbuf[stringlen - 1] = '\0';
end_process = strcmp(readbuf, end_str);
if (end_process != 0) {
write(fd, readbuf, strlen(readbuf));
printf("Sent string: \"%s\" and string length is %d\n",readbuf, (int)strlen(readbuf));
} else {
write(fd, readbuf, strlen(readbuf));
printf("Sent string: \"%s\" and string length is %d\n", readbuf, (int)strlen(readbuf));
close(fd);
break;
}}
return 0;
}
```

Output:

```
FIFO_CLIENT: Send messages, infinitely, to end enter "end"
Enter string: this is string 1
Sent string: "this is string 1" and string length is 16
Enter string: fifo test
Sent string: "fifo test" and string length is 9
Enter string: fifo client and server
Sent string: "fifo client and server" and string length is 22
Enter string: end
Sent string: "end" and string length is 3
```