BUBT | BANGLADESH UNIVERSITY OF BUSINESS AND TECHNOLOGY

Committed to Academic Excellence

Lab Report on

Array, string, sorting, searching, linked list, stack, queue, graph

Course Code: CSE 232

Course Title: Data Structures

Submitted to:	Submitted by:
Name: Suman Saha	Name: Syeda Nowshin Ibnat
Assistant Professor, Dept. of	ID: 17183103020
Computer Science & Engineering	Intake: 39
at Bangladesh University of Business and Technology.	Section: 1
	Program: B.SC. in CSE

Date of Submission: 15.09.19

<u>**Lab-1(2D array multiplication, sorting numbers)**</u>

Objective:

- The objective of this lab is to implement Array.

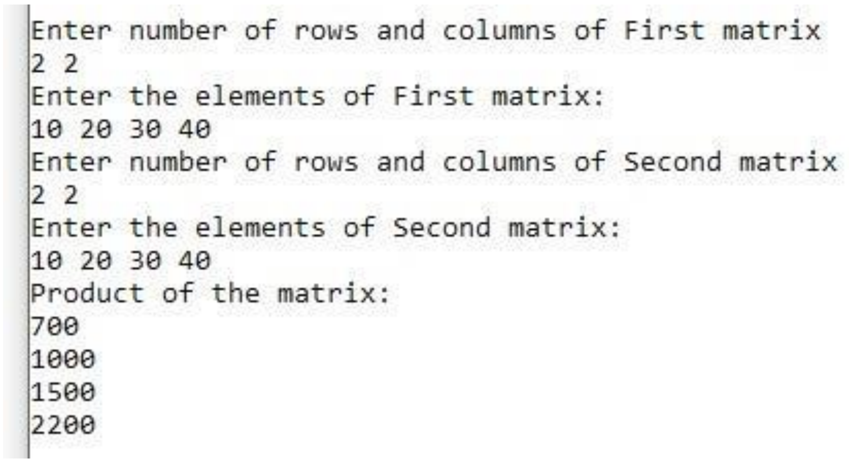
Prog-1

Title: A program to implement 2D array multiplication.

Source Code:

```
#include<stdio.h>
int main() {
int m,n,p,q,c,d,k,sum=0;
int first[10][10], second[10][10],multiply[10][10];
printf("Enter number of rows and columns of First matrix\n");
scanf("%d%d",&m,&n);
printf("Enter the elements of First matrix: \n");
for(c=0;c<m;c++)
for(d=0;d<n;d++)
scanf("%d",&first[c][d]);
printf("Enter number of rows and columns of Second matrix\n");
scanf("%d%d",&p,&q);
if(n!=p)
printf("The matrix can't be multiplied with each other.\n");
else{
printf("Enter the elements of Second matrix: \n");
for(c=0;c<p;c++)
for(d=0;d<q;d++)
scanf("%d",&second[c][d]);
for(c=0;c<m;c++){
for(d=0;d<q;d++){
for(k=0;k<p;k++){
sum=sum+(first[c][k]*second[k][d]);
multiply[c][d]=sum; }
sum=0; } }
printf("Product of the matrix: \n");
for(c=0;c<m;c++){
for(d=0;d<q;d++){
printf("%d\t",multiply[c][d]);
printf("\n"); } } }

return 0; }
```

Output:



```
Enter number of rows and columns of First matrix
2 2
Enter the elements of First matrix:
10 20 30 40
Enter number of rows and columns of Second matrix
2 2
Enter the elements of Second matrix:
10 20 30 40
Product of the matrix:
700
1000
1500
2200
```

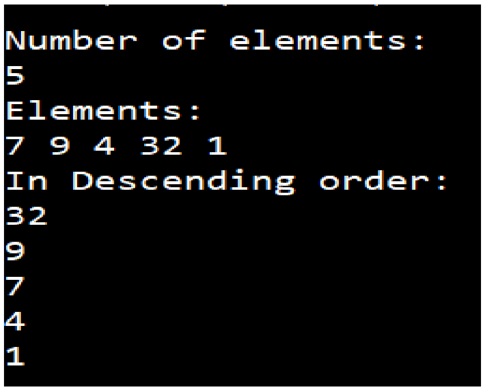
Prog-2

Title: A program to sort numbers in descending order of an array.

Source Code:

```
#include<stdio.h>

int main()  {

int i,j,n,a[50],temp;

printf("Number of elements:\n");

scanf("%d",&n);

printf("Elements:\n");

for(i=0;i<n;i++) {  scanf("%d",&a[i]); }
for(i=0;i<n;i++) { for(j=i+1;j<n;j++) {

if(a[i]<a[j]) {

temp=a[i];

a[i]=a[j];

a[j]=temp; } } }

printf(" In Descending order :\n");

for(i=0;i<n;i++) {  printf("%d\n",a[i]); }

return 0; }
```

Sample Output:



```
Number of elements:
5
Elements:
7 9 4 32 1
In Descending order:
32
9
7
4
1
```

Lab-2(String related program)

Objective:

- The objective of this lab to do string program.

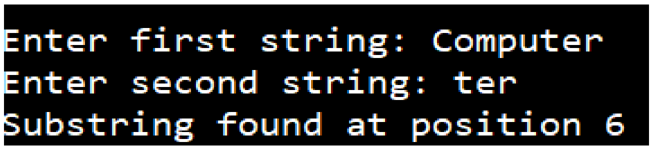
Prog-3

Title: A program where we have to take a string then store it in two different array. Then check one character whether it is existing in the string.

Source Code:

```
#include <stdio.h>

int main() {

char str1[80], str2[80];

int l, i, j;

printf("Enter first string: ");

gets(str1);

printf("Enter second string: ");

gets(str2);

for (l = 0; str2[l] != '\0'; l++);

for (i = 0, j = 0; str1[i] != '\0' && str2[j] != '\0'; i++) {

if (str1[i] == str2[j]) {  j++; }

else { j = 0;} }  if (j == l) {

printf("Substring found at position %d", i - j + 1); }

else { printf("Substring not found")

return 0; }
```

Sample Output:



```
Enter first string: Computer
Enter second string: ter
Substring found at position 6
```

Lab-3(String related programs)

Objective:

- The objective of this lab to do different types of string programs.

Prog-4

Title: A program to insert a pattern into a text.

Source Code:

```
#include <stdio.h>

#include <conio.h>

#include <string.h>

int main() {

char a[10];

char b[10];

char c[10];

int p=0,r=0,i=0;

int t=0;

int x,g,s,n,o;

puts("Enter String:");

gets(a);

puts("Enter Substring:");

 gets(b);

 printf("Enter the position:");

 scanf("%d",&p);

 r = strlen(a);

 n = strlen(b);  i=0;
```

 while(i <= r) { c[i]=a[i];

i++; } s = n+r;

o = p+n;

for(i=p; i<s; i++) { x = c[i];

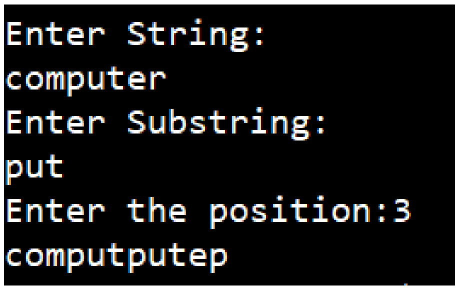
if(t<n) { a[i] = b[t];

t=t+1; } a[o]=x;

o=o+1; } printf("%s", a);

return 0; }

Sample Output:



```
Enter String:
computer
Enter Substring:
put
Enter the position:3
computputep
```

Prog-5

Title: A program to find a substring of a text.

Source Code:

#include <stdio.h>

#include <string.h>

int match(char [], char []);

int main() {

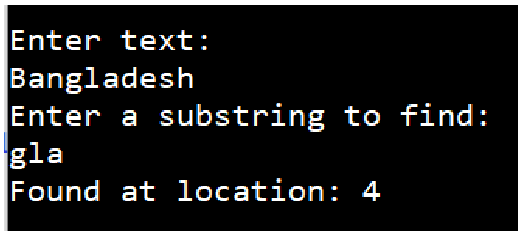
char a[100], b[100];

int position;

printf("Enter some text\n");

```
gets(a);

printf("Enter a string to find\n");

gets(b);

position = match(a, b);

if (position != -1) {

printf("Found at location: %d\n", position + 1); }

else { printf("Not found.\n"); }

return 0; }

int match(char text[], char pattern[]) {

int c, d, e, text_length, pattern_length, position = -1;

text_length    = strlen(text);

pattern_length = strlen(pattern);

if (pattern_length > text_length) { return -1; }

for (c = 0; c <= text_length - pattern_length; c++) {

position = e = c;

for (d = 0; d < pattern_length; d++) {

if (pattern[d] == text[e]) {   e++; } else {

break; } }

if (d == pattern_length) {

return position; } } return -1;}
```

Sample Output:



```
Enter text:
Bangladesh
Enter a substring to find:
gla
Found at location: 4
```

Lab-3(String related programs)

Objective:

- The objective of this lab to do string related programs.

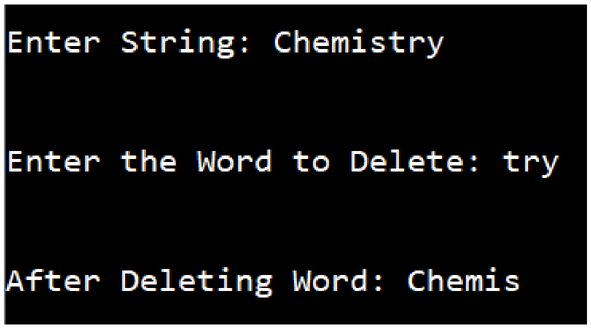
Prog-6

Title: A program to delete a pattern from a text.

Source Code:

```
#include<stdio.h>

#include<string.h>

void main() {

int i, j = 0, k = 0,n = 0;

int flag = 0;

char str[100], new[100], word[100];

printf("\nEnter Any String: ");

gets(str);

printf("\n\nEnter Any Word You Want to be Removed: ");

gets(word)

for(i = 0 ; str[i] != '\0' ; i++) {

k = i;

while(str[i] == word[j]) {

i++,j++;

if(j == strlen(word)){

flag = 1;

break; } }

j = 0;

if(flag == 0)
i = k;

else

flag = 0;

new[n++] = str[i]; }

new[n] = '\0';

printf("\n\nAfter Removing Word From String: %s\n",new);

return 0; }
```

Sample Output:



```
Enter String: Chemistry


Enter the Word to Delete: try


After Deleting Word: Chemis
```

Prog-7

Title: A program to replace a pattern from a text.

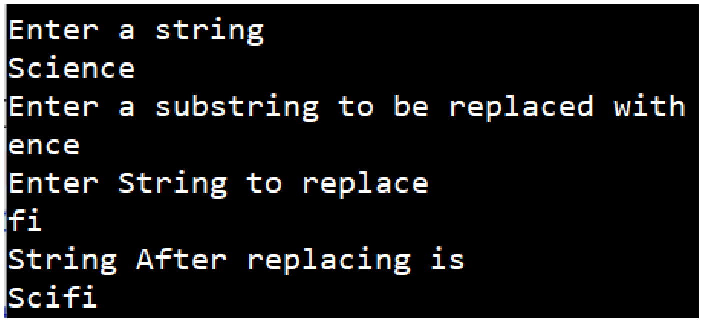
Source Code:

```
#include<stdio.h>

#include<string.h>

main() {

int i,j,flag=0,len1,len2,replaceLen,start,end;

char str[100],substr[100],replace[20],temp[120];

printf("Enter a string\n");

gets(str);
```

```
printf("Enter a substring to be replaced with\n");

gets(substr);

printf("Enter String to replace\n");

gets(replace);

len1=strlen(str);

len2=strlen(substr);

replaceLen=strlen(replace);

strcpy(temp,str);

for(i=0;i<=len1-len2;i++) {

start=i;

for(j=i;j<i+len2;j++) {

flag=1;

if(str[j]!=substr[j-i]) {

flag=0;

break; }

else{

end=j; } }

if(flag==1)

break; }

if(flag==1) {

int k=0;

if(len2==replaceLen){

for(i=0;i<len1;i++) {

if(i>=start && i<=end) {

str[i]=replace[k];
k++; } } }

if(replaceLen-len2>0) {

for(i=0;i<len1;i++) {

if(i>=start && i<=end) {

str[i]=replace[k];

k++; } }

k=0;

int x;

x=end+1;

for(i=0;i<len1+(replaceLen-len2);i++) {

if(i<=end) {

temp[i]=str[i]; }

if(i>end && i<=end+(replaceLen-len2)) {

temp[i]=replace[len2+k];

k++; }

if(i>end+(replaceLen-len2)) {

temp[i]=str[x++]; } }

temp[i]='\0';

strcpy(str,temp); }

if(replaceLen-len2<0) {

int rem=end+(replaceLen-len2);

for(i=0;i<len1;i++) {

if(i>=start && i<=rem) {

str[i]=replace[k];

k++; } }
```

```
k=0;

int x;

x=end+1;

for(i=0;i<len1+(replaceLen-len2);i++) {

if(i<=end) {

temp[i]=str[i]; }

if(i>end+(replaceLen-len2)) {

temp[i]=str[x++]; } }

temp[i]='\0';

strcpy(str,temp); }

printf("String After replacing is\n%s\n",str); }

else {  printf("Entered Substring not Found\n"); } }
```

Sample Output:



```
Enter a string
Science
Enter a substring to be replaced with
ence
Enter String to replace
fi
String After replacing is
Scifi
```

<u>**Lab-5(Sorting and searching)**</u>

Objective:

- The objective of this lab is to do different types of sorting and searching programs.

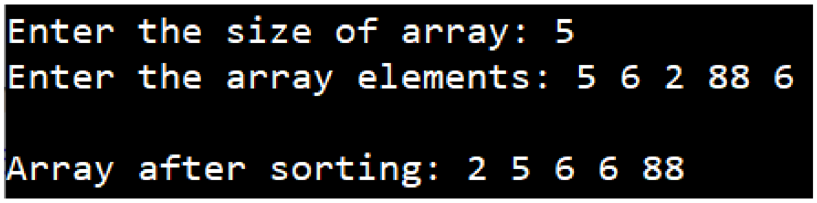
Prog-8

Title: A program to sort numbers using bubble sort.

Source Code:

```
#include<stdio.h>

int main() {
int a[50],n,i,j,temp;
printf("Enter the size of array: ");
scanf("%d",&n);
printf("Enter the array elements: ");
for(i=0;i<n;++i)
scanf("%d",&a[i]);
for(i=1;i<n;++i)
for(j=0;j<(n-i);++j)
if(a[j]>a[j+1]) {
temp=a[j];
a[j]=a[j+1];
a[j+1]=temp; }
printf("\nArray after sorting: ");
for(i=0;i<n;++i)
printf("%d ",a[i]);
return 0;}
```

Sample Output:



```
Enter the size of array: 5
Enter the array elements: 5 6 2 88 6

Array after sorting: 2 5 6 6 88
```

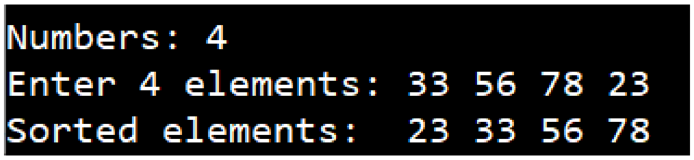
Prog-9

Title: A program to sort numbers using insertion sort.

Source Code:

```
#include<stdio.h>

int main(){

int i, j, count, temp, number[25];
printf("Numbers: ");
scanf("%d",&count);
printf("Enter %d elements: ", count);
for(i=0;i<count;i++)
scanf("%d",&number[i]);
for(i=1;i<count;i++){
temp=number[i];  j=i-1;
while((temp<number[j])&&(j>=0)){
number[j+1]=number[j];  j=j-1; }
number[j+1]=temp; }
printf("Sorted elements: ");
for(i=0;i<count;i++)
printf(" %d",number[i]);
return 0; }
```

Sample Output:



```
Numbers: 4
Enter 4 elements: 33 56 78 23
Sorted elements:  23 33 56 78
```

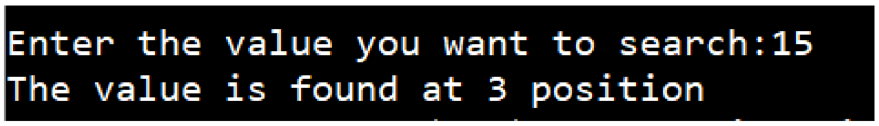
Prog-10

Title: A program to implement linear search.

Source Code:.

```
#include<stdio.h>

int main() {

int num[] = {10,2,15,20,35,46,85};

int value,pos=-1,i;

printf("Enter the value you want to search:");

scanf("%d",&value);

for(i=0; i<7; i++) {

if(value==num[i]) {

pos = i+1;

break; } }

if(pos==-1) {

printf("Item is not found"); }

else {

printf("The value is found at %d position",pos); }

return 0; }
```

Sample Output:



```
Enter the value you want to search:15
The value is found at 3 position
```

Prog-11

Title: A program to implement binary search.

Source Code:.

```
#include <stdio.h>

int main() {

int c, first, last, middle, n, search, array[100];

printf("Enter number of elements:\n");

scanf("%d",&n);

printf("Enter %d integers:\n", n);

for (c = 0; c < n; c++)

scanf("%d",&array[c]);

printf("Enter value to find:\n");

scanf("%d", &search);

first = 0;

last = n - 1;

middle = (first+last)/2;

while (first <= last) {

if (array[middle] < search)

first = middle + 1;

else if (array[middle] == search) {

printf("%d found at location %d.\n", search, middle+1);
```

break; }

else

last = middle - 1;

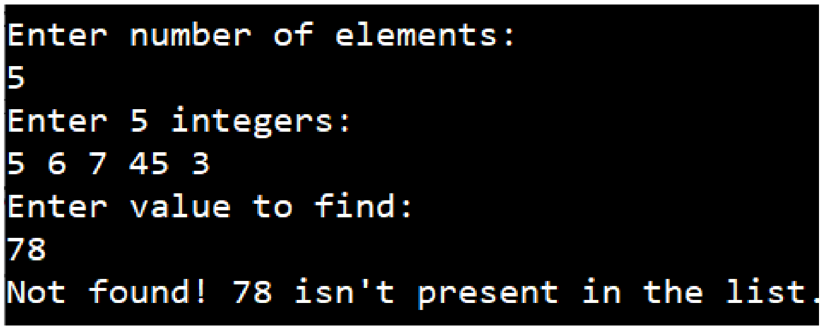
middle = (first + last)/2; }

if (first > last)

printf("Not found! %d isn't present in the list.\n", search);

return 0; }

Sample Output:



```
Enter number of elements:
5
Enter 5 integers:
5 6 7 45 3
Enter value to find:
78
Not found! 78 isn't present in the list.
```

Lab-6(Linked list)

Objective:

- The objective of this lab is to do linked list implementation.

Prog-12

Title: A program to create a simple linked list.

Source Code:

```
#include<stdio.h>
struct node {
int value;
struct node* next;};
int main(){
struct node a,b,c;
struct node* i;
struct node addfirst,addlast;
int item;
a.value=100;
b.value=200;
c.value=300;
a.next=&b;
b.next=&c;
c.next=NULL;
for(i=&a;i!=NULL;i=i->next)
printf("%d ",i->value);
return 0;}
```

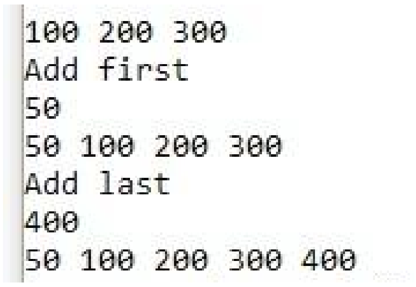
Prog-13

Title: A program to insert a node into Linked list.

Source Code:

```
#include<stdio.h>
struct node {
int value;
struct node* next; };
int main() {
struct node a,b,c;
struct node* i;
struct node addfirst,addlast;
int item;
a.value=100;
b.value=200;
c.value=300;

a.next=&b;
b.next=&c;
c.next=NULL;
for(i=&a;i!=NULL;i=i->next)
printf("%d ",i->value);
printf("\nAdd first\n");
scanf("%d",&item);
addfirst.value=item;
addfirst.next=&a;
for(i=&addfirst;i!=NULL;i=i->next)
printf("%d ",i->value);
printf("\nAdd last\n");
scanf("%d",&item);
c.next=&addlast;
addlast.value=item;
addlast.next=NULL;
for(i=&addfirst;i!=NULL;i=i->next)
printf("%d ",i->value);
return 0; }
```

Sample Output:



```
100 200 300
Add first
50
50 100 200 300
Add last
400
50 100 200 300 400
```

Lab-7(Stack)

Objective:

- The objective of this lab is to do Stack implementation.

Prog-14

Title: A program to do Stack implementation using array.

Operations are:

I. Push
II. Pop
III. Display
IV. Exit

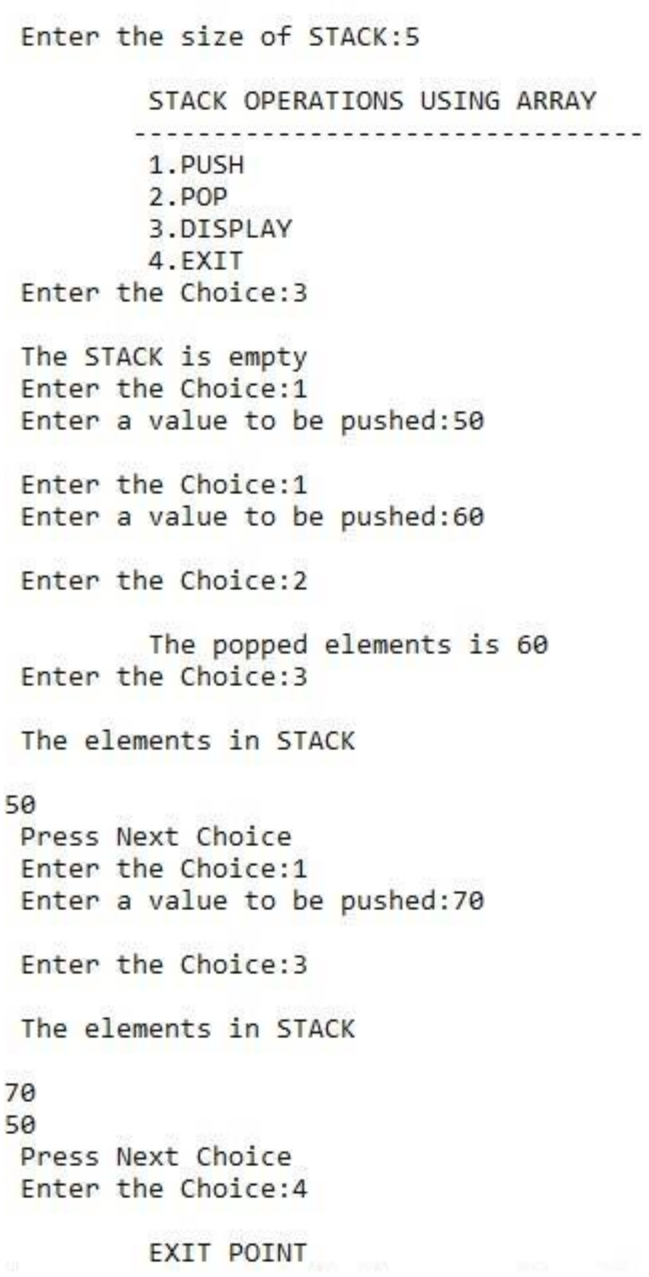
Source Code:

```
#include<stdio.h>
int stack[100],choice,n,top,x,i;
void push(void);
void pop(void);
void display(void);
int main() {
top=-1;
printf("\n Enter the size of STACK:");
scanf("%d",&n);
printf("\n\t STACK OPERATIONS USING ARRAY");
printf("\n\t--------------------------------");
printf("\n\t 1.PUSH\n\t 2.POP\n\t 3.DISPLAY\n\t 4.EXIT");
do {
printf("\n Enter the Choice:");
scanf("%d",&choice);
switch(choice) {
case 1: {
```

```
push();
break; }
case 2: {
pop();
break; }
case 3: {
display();
break; }
case 4: {
printf("\n\t EXIT POINT ");
break; }
default: {
printf ("\n\t Please Enter a Valid Choice(1/2/3/4)"); } } }
while(choice!=4);
return 0; }
void push() {
if(top>=n-1) {
printf("\n\tSTACK is over flow"); }
else {
printf(" Enter a value to be pushed:");
scanf("%d",&x);
top++;
stack[top]=x; } }
void pop() {
if(top<=-1) {
printf("\n\t Stack is under flow"); }
else {
printf("\n\t The popped elements is %d",stack[top]);
top--; } }
void display() {
if(top>=0) {
```

```
printf("\n The elements in STACK \n");
for(i=top; i>=0; i--)
printf("\n%d",stack[i]);
printf("\n Press Next Choice"); }
else {
printf("\n The STACK is empty"); } }
```

Sample Output:



```
 Enter the size of STACK:5

         STACK OPERATIONS USING ARRAY
        --------------------------------
         1.PUSH
         2.POP
         3.DISPLAY
         4.EXIT
 Enter the Choice:3

 The STACK is empty
 Enter the Choice:1
 Enter a value to be pushed:50

 Enter the Choice:1
 Enter a value to be pushed:60

 Enter the Choice:2

         The popped elements is 60
 Enter the Choice:3

 The elements in STACK

50
 Press Next Choice
 Enter the Choice:1
 Enter a value to be pushed:70

 Enter the Choice:3

 The elements in STACK

70
50
 Press Next Choice
 Enter the Choice:4

         EXIT POINT
```

Lab-8(Queue)

Prog-15

Objective:

- The objective of this lab is to do Queue implementation.

Title: A program to of Queue implementation.

Operations are:

I. Insert ()
II. Delete ()
III. Is empty ()
IV. Is full ()
V. Display the front element
VI. Exit.

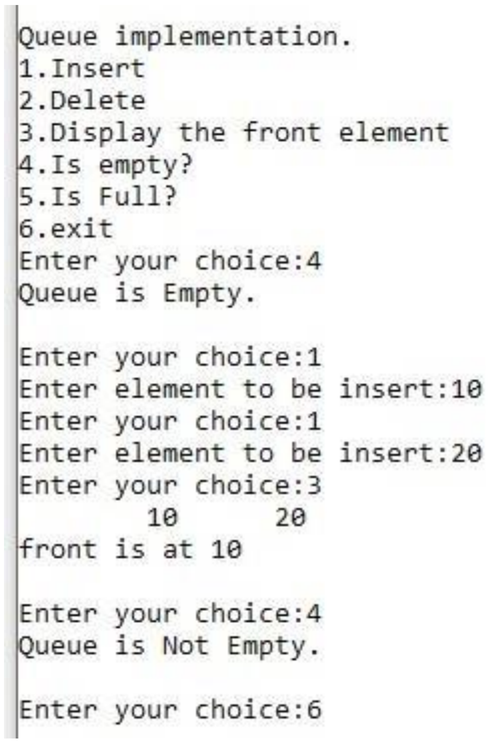
Source Code:

```
#include<stdio.h>
#include<stdlib.h>
#define max 5
int q[max],front=0,rear=-1;
void main() {
int ch;
void insert();
void delet();
void display();
void isempty();
void isfull();
printf("\nQueue implementation.\n");
printf("1.Insert\n2.Delete\n3.Display the front element\n4.Is
empty? \n5.Is Full?\n6.exit\n");
while(1) {
printf("Enter your choice:");
scanf("%d",&ch);
switch(ch) {
case 1: insert();
break;
case 2: delet();
break;
case 3:display();
break;
case 4:
isempty();
break;
case 5:
isfull();
break;
case 6:
exit(0);
default:printf("Invalid option\n"); } }
return 0; }
void insert() {
int x;
if((front==0&&rear==max-1)||(front>0&&rear==front-1))
printf("Queue is overflow\n");
else {
printf("Enter element to be insert:");
scanf("%d",&x);
if(rear==max-1&&front>0) {
rear=0;
q[rear]=x; }
else {
if((front==0&&rear==-1)||(rear!=front-1))
q[++rear]=x; } } }
void delet() {
```

```
int a;
if((front==0)&&(rear==-1)) {
printf("Queue is underflow\n");
exit(0); }
if(front==rear) {
a=q[front];
rear=-1;
front=0; }
else
if(front==max-1) {
a=q[front];
front=0; }
else a=q[front++];
printf("Deleted element is:%d\n",a); }

void display() {
int i,j;
if(front==0&&rear==-1) {
printf("Queue is underflow\n");
exit(0); }
if(front>rear) {
for(i=0;i<=rear;i++)
printf("\t%d",q[i]);
for(j=front;j<=max-1;j++)
printf("\t%d",q[j]);
printf("\nfront is at %d\n",q[front]); }
else {
for(i=front;i<=rear;i++) {
printf("\t%d",q[i]); }
printf("\nfront is at %d\n",q[front]); }
printf("\n"); }
void isfull(){
```

```
if((front==0&&rear==max-1)||(front>0&&rear==front-1))
printf("Queue is Full\n\n");
else
printf("Queue is Not Full.\n\n"); }
void isempty(){
if(front==0&&rear==-1) {
printf("Queue is Empty.\n\n"); }
else
printf("Queue is Not Empty.\n\n"); }
```

Sample Output:



```
Queue implementation.
1.Insert
2.Delete
3.Display the front element
4.Is empty?
5.Is Full?
6.exit
Enter your choice:4
Queue is Empty.

Enter your choice:1
Enter element to be insert:10
Enter your choice:1
Enter element to be insert:20
Enter your choice:3
        10      20
front is at 10

Enter your choice:4
Queue is Not Empty.

Enter your choice:6
```

Lab-8(Graph)

Prog-16

Objective:

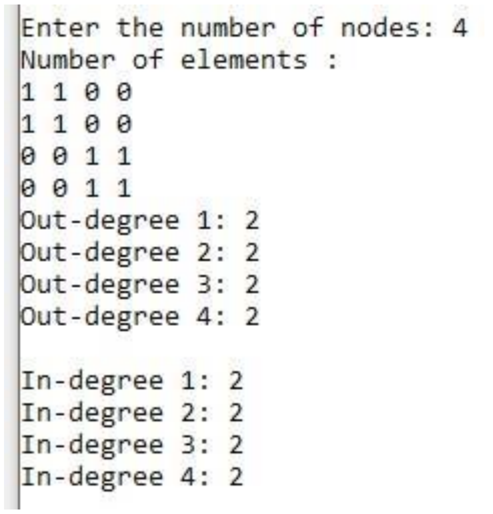
- The objective of this lab is to do Graph implementation using adjacency matrix.

Title: A program to do graph implementation.

Source Code:

```
#include<stdio.h>
int main() {
int i,j,sum=0,n,m[10][10];
printf("Enter the number of nodes: ");
scanf("%d",&n);
printf("Number of elements :\n");
for(i=0;i<n;i++)
for(j=0;j<n;j++)
scanf("%d",&m[i][j]);
for(i=0;i<n;i++){
sum=0;
for(j=0;j<n;j++){
sum=sum+m[j][i]; }
printf("Out-degree %d: %d\n",i+1,sum); }
printf("\n");
for(i=0;i<n;i++){
sum=0;
for(j=0;j<n;j++){
sum=sum+m[i][j]; }
printf("In-degree %d: %d\n",i+1,sum); }
return 0; }
```

Sample Output:



```
Enter the number of nodes: 4
Number of elements :
1 1 0 0
1 1 0 0
0 0 1 1
0 0 1 1
Out-degree 1: 2
Out-degree 2: 2
Out-degree 3: 2
Out-degree 4: 2

In-degree 1: 2
In-degree 2: 2
In-degree 3: 2
In-degree 4: 2
```