MORADABAD INSTITUTE OF TECHNOLOGY

MORADABAD

MIT GROUP OF INSTITUTIONS

In Pursuit of Excellence

SESSION: 2019-20

PRACTICAL FILE

DATA STRUCTURES USING C LAB

(KCS351)

SUBMITTED TO:

MR. MANOJ KUMAR SINGH

SUBMITTED BY:

NAME: Harshit Pandey
ROLL. NO.: 1808210063
BRANCH: CSE
BATCH: A3

Dr. Somesh Kumar
Prof. & Head, CSE
Moradabad Institute of Technology
Moradabad-244001

INDEX

Dr. Somesh Kumar
Prof. & Head, CSE
Moradabad Institute of Technology
Moradabad-244001

INDEX

06/08/19 PROGRAM - 1

OBJECT: Program to multiply two matrix

Source Code:

```
#include<stdio.h>
void main()
{
int a[10][10],b[10][10],c[10][10],r1,c1,r2,c2,i,j,k;
printf(“Enter the number of rows and columns of matrix1\n”);
scanf(“%d%d”,&r1,&c1);
printf(“Enter Matrix1\n”);
for(i = 0 ; i < r1; i++)
{
for(j = 0 ; j < c1 ; j++)
{
scanf(“%d”,&a[i][j]);
}
}
printf(“Enter the number of rows and columns of matrix2\n”);
scanf(“%d”,&b[i][j]);
printf(“Enter Matrix2\n”);
for(i = 0 ; i < r1 ; i++)
{
for(j = 0; j < c1 ; j++)
{
scanf(“%d”,&a[i][j]);
}
}
if(c1 != r2)
{
printf(“Product is not possible\n”);
}
else
{
for(i = 0 ; i < r1 ; i++)
{
for(j = 0 ; j < c2 ; j++)
{
prod=0;
for(k = 0 ; k < c1 ; k++)
{
prod = prod + a[i][k] * b[k][j];
}
c[i][j] = prod;
}
}
printf(“Product Matrix is\n”);
for(i = 0 ; i < r1 ; i++)
{
for(j = 0 ; j < c2 ; j++)
{
printf(“%d\t”,c[i][j]);
}
printf(“\n”);
}
}
}
```

Dr. Somesh Kumar
Prof. & Head, CSE
Moradabad Institute of Technology
Moradabad-244001

Output:

```
Enter the number of rows and columns of matrix1
3
3
Enter Matrix1
1
2
3
4
5
6
7
8
9
Enter the number of rows and columns of matrix2
3
3
Enter Matrix2
1
2
3
4
5
6
7
8
9
Product Matrix is
30      36      42
66      81      96
102     126     150
```

13/08/19

06/08/19

PROGRAM - 2

OBJECT: Program to implement Linear Search

Source Code:

```
#include<stdio.h>
void main()
{
int a[100],n,i,s,flag=0;
printf("Enter the size of array\n");
scanf("%d",&n);
printf("Enter the array\n");
for(i = 0 ; i < n ; i++)
{
scanf("%d",&a[i]);
}
printf("Enter the element to be searched\n");
scanf("%d",&s);
for(i = 0 ; i < n ; i++)
{
if(a[i] == s)
{
flag = 1;
break;
}
}
if(flag == 1)
{
printf("%d is present at %d index\n",s,i);
}
else
{
printf("%d is not present\n",s);
}
}
```

Dr. Somesh Kumar
Prof. & Head, CSE
Moradabad Institute of Technology
Moradabad-244001

Output:

```
Enter the size of array
10
Enter the array
1
2
3
4
5
6
7
8
9
10
Enter the element to be searched
8
8 is present at index 7
```

A+

13/08/19

Dr. Somesh Kumar
Prof. & Head, CSE
Moradabad Institute of Technology
Moradabad-244001

13/08/19

PROGRAM - 3

OBJECT: Program to implement Binary Search

Source Code:

```
#include<stdio.h>
void main()
{
Int a[100],i,middle,first,last,n,search;
printf("Enter the size of array\n");
scanf("%d",&n);
printf("Enter the elements of array\n");
for(i = 0; i < n; i++)
{
scanf("%d",&a[i]);
}
printf("Enter the element to be searched\n");
scanf("%d",&search);
first = 0;
last = n-1;
middle = (first+last)/2;
while(first<=last)
{
if(a[middle]<search)
{
first = middle +1;
}
else if(a[middle]==search)
{
printf("%d is present at index %d",s,middle);
break;
}
else
{
last = middle - 1;
}
middle = (first+last)/2;
}
if(first>last)
{
printf("Element not present\n");
}
}
```

Dr. Somesh Kumar
Prof. & Head, CSE
Moradabad Institute of Technology
Moradabad-244001

Output:

```
Enter the size of array
10
Enter the array
1
2
3
4
5
6
7
8
9
10
Enter the element to be searched
8
8 is present at index 7
```

A+

20/08/19

Dr. Somesh Kumar
Prof. & Head, CSE
Moradabad Institute of Technology
Moradabad-244001

20/08/19

PROGRAM - 4

OBJECT: Program to implement Selection Sort

Source Code:

```
#include<stdio.h>
void main()
{
int a[100],n,i,j,min,t;
printf(“Enter the size of array\n”);
scanf(“%d”,&n);
printf(“Enter the array\n”);
for(i = 0 ; i < n ; i++)
{
scanf(“%d”,&a[i]);
}
for(i = 0 ; i < n-1 ; i++)
{
min = i;
for(j = i+1 ; j < n ; j++)
{
if(a[j]<a[min])
{
min = j;
}
t = a[min];
a[min] = a[i];
a[i] = t;
}
}
printf(“Sorted array is\n”);
for(i = 0 ; i < n ; i++)
{
printf(“%d\t”,a[i]);
}
}
```

Dr. Somesh Kumar
Prof. & Head, CSE
Moradabad Institute of Technology
Moradabad

Output:

```
Enter the size of array
10
Enter the array
10
9
8
7
6
5
4
3
2
1
Sorted array is
1       2       3       4       5       6       7       8       9       10
```

27

27/08/19

27/08/19

PROGRAM - 5

OBJECT: Program to implement Bubble Sort

Source Code:

```
#include<stdio.h>
void main()
{
int a[100],i,j,n,t;
printf(“Enter the size of array\n”);
scanf(“%d”,&n);
printf(“Enter the array\n”);
for(i = 0 ; i < n ; i++)
{
scanf(“%d”,&a[i]);
}
for(i = 0 ; i < n-1 ; i++)
{
for(j = 0 ; j < n-i-1 ; j++)
{
if(a[j]>a[j+1])
{
t = a[j];
a[j] = a[j+1];
a[j+1] = t;
}
}
}
printf(“Sorted array is\n”);
for(i = 0 ; i < n ; i++)
{
printf(“%d\t”,a[i]);
}
}
```

Dr. Somesh Kumar
Prof. & Head, CSE
Moradabad Institute of Technology
Moradabad-244001

Output:

```
Enter the size of array
10
Enter the array
10
9
8
7
6
5
4
3
2
1
Sorted array is
1       2       3       4       5       6       7       8       9       10
```

27

27/08/19

Dr. Somesh Kumar
Prof. & Head, CSE
Moradabad Institute of Technology
Moradabad-244001

27/08/19

PROGRAM - 6

OBJECT: Program to implement Insertion Sort

Source Code:

```
#include<stdio.h>
void main()
{
int a[100],n,i,j,key;
printf(“Enter the size of array\n”);
scanf(“%d”,&n);
printf(“Enter the array\n”);
for(i = 0 ; i < n ; i++)
{
scanf(“%d”,&a[i]);
}
for(i = 1 ; i < n ; i++)
{
key = a[i];
j = i - 1;
{
while(j>=0 && a[j]>key)
{
a[j+1] = key;
j = j-1;
}
a[j+1] = key;
}
printf(“Sorted array is\n”);
for(i = 0 ; i < n ; i++)
{
printf(“%d\t”,a[i]);
}
}
```

Dr. Somesh Kumar
Prof. & Head, CSE
Moradabad Institute of Technology
Moradabad-244001

Output:

```
Enter the size of array
10
Enter the array
10
9
8
7
6
5
4
3
2
1
Sorted array is
1       2       3       4       5       6       7       8       9       10
```

A+

03/09/19

Dr. Somesh Kumar
Prof. & Head, CSE
Moradabad Institute of Technology
Moradabad-244001

03/09/19

PROGRAM - 7

OBJECT: Program to implement Quick Sort

Source Code:

```
#include<stdio.h>
int partition(int a[] , int p , int r)
{
int i , j , x , temp;
x = a[p];
i = p - 1;
j= r + 1;
while(1)
{
do
{
j--;
}while(a[j] > x);
do
{
i++;
}while(a[i] < x);
if(i < j)
{
temp = a[i];
a[i] = a[j];
a[j] = temp;
}
else
return j;
}
}
void quicksort(int a[] , int p , int r)
{
int q;
if(p < r)
{
q = partition(a , p , r);
quicksort(a , p , q);
quicksort(a , q+1 , r);
}
}
void main()
{
int a[100] , i , n;
printf("Enter the size of array\n");
scanf("%d",&n);
printf("Enter array\n");
for(i = 0 ; i < n ; i++)
{
scanf("%d",&a[i]);
}
quicksort(a , 0 , n-1);
printf("Sorted array is\n");
for(i = 0 ; i < n ; i++)
{
printf("%d ",a[i]);
}}
```

Dr. Somesh Kumar
Prof. & Head, CSE
Moradabad Institute of Technology
Moradabad-244001

03/09/19

PROGRAM - 7

Output:

```
Enter the size of array
10
Enter array
10
9
8
7
6
5
4
3
2
1
Sorted array is
1	2	3	4	5	6	7	8	9	10
```

17/09/19

PROGRAM - 8

OBJECT: Program to implement Stack using Array

Source Code:

```
#include<stdio.h>
#include<stdlib.h>
int stack[20];
int n;
int top = -1;
void push(int item)
{
if(top == n-1)
{
printf("Stack Overflow\n");
return;
}
top = top+1;
stack[top] = item;
}
void pop()
{
int x;
if(top==-1)
{
printf("Stack Underflow\n");
return;
}
x = stack[top];
top = top - 1;
printf("\nElement deleted is: %d\n",x);
}
void display()
{
int i;
if(top==-1)
{
printf("Stack is empty\n");
return;
}
for(i=top;i>=0;i--)
{
printf("%d ",stack[i]);
}
printf("\n");
}
void main()
{
int m,item;
printf("Enter the value of n: ");
scanf("%d",&n);
do
{
printf("\n1: Push");
printf("\n2: Pop");
printf("\n3: Display");
printf("\n4: Exit");
printf("\nEnter your choice: ");
scanf("%d",&m);
switch(m)
{
case 1:printf("Enter the element: ");
scanf("%d",&item);
push(item);
break;
case 2:pop();
break;
case 3:display();
break;
case 4:printf(“Program Ended\n”);
break;
default:printf("Wrong Choice\n");
}
}while(m!=4);
}
```

Output:

```
Enter the value of n: 5

1: Push
2: Pop
3: Display
4: Exit
Enter your choice: 1
Enter the element: 1

1: Push
2: Pop
3: Display
4: Exit
Enter your choice: 1
Enter the element: 2

1: Push
2: Pop
3: Display
4: Exit
Enter your choice: 1
Enter the element: 3

1: Push
2: Pop
3: Display
4: Exit
Enter your choice: 1
Enter the element: 4

1: Push
2: Pop
3: Display
4: Exit
Enter your choice: 3
4 3 2 1

1: Push
2: Pop
3: Display
4: Exit
Enter your choice: 2
Element deleted is: 4

1: Push
2: Pop
3: Display
4: Exit
Enter your choice: 3
3 2 1

1: Push
2: Pop
3: Display
4: Exit
Enter your choice: 4
Program Ended
```

2

15/10/19

01/10/19

PROGRAM - 9

OBJECT: Program to implement Queue using Array

Source Code:

```
#include <stdio.h>
#include <stdlib.h>
struct Queue
{
int size;
int front;
int rear;
int *Q;
};
void Create(struct Queue *q,int size)
{
q->size = size;
q->front = q->rear = -1;
q->Q = (int*)malloc(sizeof(int)*q->size);
}
void Enqueue(struct Queue *q,int x)
{
if(q->rear == q->size-1)
printf("Queue is full\n");
else
{
q->rear++;
q->Q[q->rear] = x;
}
}
int Dequeue(struct Queue *q)
{
int x = -1;
if(q->front == q->rear)
{
printf("Queue is empty\n");
printf("Nothing Deleted\n");
}
else
{
q->front++;
x = q->Q[q->front];
printf("%d is deleted\n",x);
}
return x;
}
void Display(struct Queue q)
{
int i;
for(i=q.front+1;i<=q.rear;i++)
printf("%d ",q.Q[i]);
printf("\n");
}
void main()
{
struct Queue Q;
int size,m,x;
printf("Enter the size of Queue: ");
scanf("%d",&size);
Create(&Q,size);
do
{
printf(“\n”);
printf("Press 1 to Enqueue\n");
printf("Press 2 to Dequeue\n");
printf("Press 3 to Display\n");
printf("Press 4 to Exit\n");
printf("Enter the choice: ");
scanf("%d",&m);
switch(m)
{
case 1:printf("Enter the element to be inserted: ");
scanf("%d",&x);
Enqueue(&Q,x);
break;
case 2:Dequeue(&Q);
break;
case 3:Display(Q);
break;
case 4:printf("Program Ended\n");
break;
default:printf("Wrong Choice\n");
break;
}
}while(m!=4);
}
```

Output:

```
Enter the size of Queue: 5

Press 1 to Enqueue
Press 2 to Dequeue
Press 3 to Display
Press 4 to Exit
Enter the choice: 1
Enter the element to be inserted: 1

Press 1 to Enqueue
Press 2 to Dequeue
Press 3 to Display
Press 4 to Exit
Enter the choice: 1
Enter the element to be inserted: 2

Press 1 to Enqueue
Press 2 to Dequeue
Press 3 to Display
Press 4 to Exit
Enter the choice: 1
Enter the element to be inserted: 3

Press 1 to Enqueue
Press 2 to Dequeue
Press 3 to Display
Press 4 to Exit
Enter the choice: 3
1 2 3

Press 1 to Enqueue
Press 2 to Dequeue
Press 3 to Display
Press 4 to Exit
Enter the choice: 2
1 is deleted

Press 1 to Enqueue
Press 2 to Dequeue
Press 3 to Display
Press 4 to Exit
Enter the choice: 3
2 3

Press 1 to Enqueue
Press 2 to Dequeue
Press 3 to Display
Press 4 to Exit
Enter the choice: 4
Program Ended
```

2

15/10/19

Dr. Somesh Kumar
Prof. & Head, CSE
Moradabad Institute of Technology
Moradabad-244001

01/10/19

PROGRAM - 10

OBJECT: Program to implement Circular Queue using Array

Source Code:

```
#include <stdio.h>
#include <stdlib.h>
struct Queue
{
int size;
int front;
int rear;
int *Q;
};
void Create(struct Queue *q,int size)
{
q->size = size;
q->front = q->rear = 0;
q->Q = (int*)malloc(sizeof(int)*q->size);
}
void Enqueue(struct Queue *q,int x)
{
if((q->rear+1)%q->size == q->front)
printf("Queue is full\n");
else
{
q->rear = (q->rear+1)%q->size;
q->Q[q->rear] = x;
}
}
int Dequeue(struct Queue *q)
{
int x = -1;
if(q->front == q->rear)
{
printf("Queue is empty\n");
printf("Nothing Deleted\n");
}
else
{
q->front = (q->front+1)%q->size;
x = q->Q[q->front];
printf("%d is deleted\n",x);
}
return x;
}
void Display(struct Queue q)
{
int i = (q.front+1)%q.size;
do
{
```

Dr. Somesh Kumar
Prof. & Head, CSE
Moradabad Institute of Technology
Moradabad-244001

```
printf("%d ",q.Q[i]);
i = (i+1)%q.size;
}while(i!=(q.rear+1)%q.size);
printf("\n");
}
void main()
{
struct Queue Q;
int size,m,x;
printf("Enter the size of Queue: ");
scanf("%d",&size);
Create(&Q,size);
do
{
printf(“\n”);
printf("Press 1 to Enqueue\n");
printf("Press 2 to Dequeue\n");
printf("Press 3 to Display\n");
printf("Press 4 to Exit\n");
printf("Enter the choice: ");
scanf("%d",&m);
switch(m)
{
case 1:printf("Enter the element to be inserted: ");
scanf("%d",&x);
Enqueue(&Q,x);
break;
case 2:Dequeue(&Q);
break;
case 3:Display(Q);
break;
case 4:printf("Program Ended\n");
break;
default:printf("Wrong Choice\n");
break;
}
}while(m!=4);
}
```

Output:

```
Enter the size of Queue: 5

Press 1 to Enqueue
Press 2 to Dequeue
Press 3 to Display
Press 4 to Exit
Enter the choice: 1
Enter the element to be inserted: 1

Press 1 to Enqueue
Press 2 to Dequeue
Press 3 to Display
Press 4 to Exit
Enter the choice: 1
Enter the element to be inserted: 2

Press 1 to Enqueue
Press 2 to Dequeue
Press 3 to Display
Press 4 to Exit
Enter the choice: 1
Enter the element to be inserted: 3

Press 1 to Enqueue
Press 2 to Dequeue
Press 3 to Display
Press 4 to Exit
Enter the choice: 1
Enter the element to be inserted: 4

Press 1 to Enqueue
Press 2 to Dequeue
Press 3 to Display
Press 4 to Exit
Enter the choice: 3
1 2 3 4

Press 1 to Enqueue
Press 2 to Dequeue
Press 3 to Display
Press 4 to Exit
Enter the choice: 4
Program Ended
```

15/10/19

15/10/19

PROGRAM - 11

OBJECT: Program to implement Merge Sort

Source Code:

```
#include<stdio.h>
void Merge(int A[],int l,int mid,int h)
{
int i = l , j = mid+1 , k = l;
int B[100];
while(i<=mid && j<=h)
{
if(A[i]<A[j])
B[k++] = A[i++];
else
B[k++] = A[j++];
}
while(i<=mid)
B[k++] = A[i++];
while(j<=h)
B[k++] = A[j++];
for(i=0;i<=h;i++)
A[i] = B[i];
}
void MergeSort(int A[],int l,int h)
{
int mid;
if(l<h)
{
mid = (l+h)/2;
MergeSort(A,l,mid);
MergeSort(A,mid+1,h);
Merge(A,l,mid,h);
}
}
void main()
{
int A[100],n,i;
printf("Enter the size of array\n");
scanf("%d",&n);
printf("Enter the array\n");
for(i=0;i<n;i++)
scanf("%d",&A[i]);
MergeSort(A,0,n-1);
printf("Sorted array\n");
for(i=0;i<n;i++)
printf("%d\t",A[i]);
}
```

Dr. Somesh Kumar
Prof. & Head, CSE
Moradabad Institute of Technology
Moradabad-244001

15/10/19

PROGRAM - 11

Output:

```
Enter the size of array
10
Enter the array
10
9
8
7
6
5
4
3
2
1
Sorted array
1       2       3       4       5       6       7       8       9       10
```

2+

05/11/19

PROGRAM - 12

OBJECT: Program to implement Stack using Linked List

Source Code:

```
#include<stdio.h>
#include<stdlib.h>
struct Node
{
int data;
struct Node *next;
}*top=NULL;
void Push(int x)
{
struct Node *t = NULL;
t = (struct Node*)malloc(sizeof(struct Node));
if(t==NULL)
{
printf("Stack Overflow\n");
return;
}
else
{
t->data = x;
t->next = NULL;
if(top==NULL)
top = t;
else
{
t->next = top;
top = t;
}
}
}
int Pop()
{
struct Node *p = top;
int x = -1;
if(top==NULL)
{
printf("Stack Underflow\n");
}
else
{
top = top->next;
x = p->data;
free(p);
printf("%d is Popped\n",x);
}
return x;
}
```

```
void Display()
{
struct Node *p = top;
while(p)
{
printf("%d ",p->data);
p = p->next;
}
printf("\n");
}
void main()
{
int ch,x;
do
{
printf("\n");
printf("Press 1 to Push\n");
printf("Press 2 to Pop\n");
printf("Press 3 to Display\n");
printf("Press 4 to exit\n");
printf("Enter the choice: ");
scanf("%d",&ch);
switch(ch)
{
case 1:printf("Enter the element to be Pushed: ");
scanf("%d",&x);
Push(x);
break;
case 2:Pop();
break;
case 3:Display();
break;
case 4:printf("Program Ended\n");
break;
default:printf("Wrong Entry\n");
break;
}
}while(ch!=4);
}
```

Dr. Somesh Kumar
Prof. & Head, CSE
Moradabad Institute of Technology
Moradabad-244001

Output:

```
Press 1 to Push
Press 2 to Pop
Press 3 to Display
Press 4 to exit
Enter the choice: 1
Enter the element to be Pushed: 1

Press 1 to Push
Press 2 to Pop
Press 3 to Display
Press 4 to exit
Enter the choice: 1
Enter the element to be Pushed: 2

Press 1 to Push
Press 2 to Pop
Press 3 to Display
Press 4 to exit
Enter the choice: 1
Enter the element to be Pushed: 3

Press 1 to Push
Press 2 to Pop
Press 3 to Display
Press 4 to exit
Enter the choice: 3
3 2 1

Press 1 to Push
Press 2 to Pop
Press 3 to Display
Press 4 to exit
Enter the choice: 4
Program Ended
```

05/11/19

15/10/19

PROGRAM - 13

OBJECT: Program to implement Queue using Linked List

Source Code:

```
#include<stdio.h>
#include<stdlib.h>
struct Node
{
int data;
struct Node *next;
}*front=NULL,*rear=NULL;
void Enqueue(int x)
{
struct Node *t = NULL;
t = (struct Node*)malloc(sizeof(struct Node));
if(t==NULL)
{
printf("Queue is full\n");
return;
}
else
{
t->data = x;
t->next = NULL;
if(front==NULL)
{
front = t;
rear = t;
}
else
{
rear->next = t;
rear = t;
}
}
}
int Dequeue()
{
struct Node *p = front;
int x = -1;
if(front==NULL)
{
printf("Queue is empty\n");
}
else
{
front = front->next;
x = p->data;
free(p);
printf("%d is Dequeued\n",x);
}
return x;
}
void Display()
{
struct Node *p = front;
while(p)
{
printf("%d ",p->data);
p = p->next;
}
printf("\n");
}
void main()
{
int ch,x;
do
{
printf("\n");
printf("Press 1 to Enqueue\n");
printf("Press 2 to Dequeue\n");
printf("Press 3 to Display\n");
printf("Press 4 to exit\n");
printf("Enter the choice: ");
scanf("%d",&ch);
switch(ch)
{
case 1:printf("Enter the element to be Enqueued: ");
scanf("%d",&x);
Enqueue(x);
break;
case 2:Dequeue();
break;
case 3:Display();
break;
case 4:printf("Program Ended\n");
break;
default:printf("Wrong Entry\n");
break;
}
}while(ch!=4);
}
```

15/10/19

PROGRAM - 13

Output:

```
Press 1 to Enqueue
Press 2 to Dequeue
Press 3 to Display
Press 4 to exit
Enter the choice: 1
Enter the element to be Enqueued: 1

Press 1 to Enqueue
Press 2 to Dequeue
Press 3 to Display
Press 4 to exit
Enter the choice: 1
Enter the element to be Enqueued: 2

Press 1 to Enqueue
Press 2 to Dequeue
Press 3 to Display
Press 4 to exit
Enter the choice: 1
Enter the element to be Enqueued: 3

Press 1 to Enqueue
Press 2 to Dequeue
Press 3 to Display
Press 4 to exit
Enter the choice: 1
Enter the element to be Enqueued: 4

Press 1 to Enqueue
Press 2 to Dequeue
Press 3 to Display
Press 4 to exit
Enter the choice: 3
1 2 3 4

Press 1 to Enqueue
Press 2 to Dequeue
Press 3 to Display
Press 4 to exit
Enter the choice: 4
Program Ended
```

27

05/11/19

05/11/19

PROGRAM - 14

OBJECT: Program to implement priority queue using Linked List

Source Code:

```
#include<stdio.h>
#include<stdlib.h>
typedef struct node
{
int priority;
int info;
struct node *link;
}NODE;
NODE *front = NULL;
void insert(int item,int priority)
{
NODE *tmp,*q;
tmp = (NODE*)malloc(sizeof(NODE));
tmp->info = item;
tmp->priority = priority;
if(front==NULL || priority<front->priority)
{
tmp->link = front;
front = tmp;
}
else
{
q = front;
while(q->link!=NULL && q->link->priority<=priority)
q = q->link;
tmp->link = q->link;
q->link = tmp;
}
}
void del()
{
NODE *tmp;
if(front==NULL)
printf("Queue Underflow\n");
else
{
tmp = front;
printf("Deleted item is %d\n",tmp->info);
front = front->link;
free(tmp);
}
}
void display()
{
NODE *ptr;
ptr = front;
if(front==NULL)
printf("Queue is Empty\n");
else
{
printf("Queue is :\n");
printf("Priority item\n");
while(ptr!=NULL)
{
printf("(%d,%d)\n",ptr->priority,ptr->info);
ptr = ptr->link;
}
}
printf("\n");
}
void main()
{
int choice,item,priority;
do
{
printf("1. Insert\n");
printf("2. Delete\n");
printf("3. Display\n");
printf("4. Quit\n");
printf("Enter your choice: ");
scanf("%d",&choice);
switch(choice)
{
case 1:printf("Input the item value to be added in the queue: ");
scanf("%d",&item);
printf("Enter the priority: ");
scanf("%d",&priority);
insert(item,priority);
printf("\n");
break;
case 2:del();
printf("\n");
break;
case 3:display();
break;
case 4:printf("Program Ended\n");
break;
default:printf("Wrong Choice\n");
}
}while(choice!=4);
}
```

Output:

```
1. Insert
2. Delete
3. Display
4. Quit
Enter your choice: 1
Input the item value to be added in the queue: 1
Enter the priority: 3

1. Insert
2. Delete
3. Display
4. Quit
Enter your choice: 1
Input the item value to be added in the queue: 1
Enter the priority: 6

1. Insert
2. Delete
3. Display
4. Quit
Enter your choice: 3
Queue is :
Priority item
(3,1)
(6,1)

1. Insert
2. Delete
3. Display
4. Quit
Enter your choice: 4
Program Ended
```

A

19/11/19

05/11/19

PROGRAM - 15

OBJECT: Program to implement solution of Tower of Hanoi problem using recursion

Source Code:

```
#include<stdio.h>
void TOH(int n,char A,char B,char C)
{
if(n>0)
{
TOH(n-1,A,C,B);
printf("(%c,%c)\n",A,C);
TOH(n-1,B,A,C);
}
}
void main()
{
int n;
printf("Enter the number of rings\n");
scanf("%d",&n);
printf("Moves\n");
TOH(n,'A','B','C');
}
```

Output:

```
Enter the number of rings
3
Moves
(A,C)
(A,B)
(C,B)
(A,C)
(B,A)
(B,C)
(A,C)
```

19/11/19

A+

05/11/19 PROGRAM - 16

OBJECT: Program to implement Binary Search Tree

Source Code:

```
#include<stdio.h>
#include<stdlib.h>
struct node
{
int info;
struct node *left, *right;
}*root=NULL;
void inorder(struct node *r)
{
if (r != NULL)
{
inorder(r->left);
printf("%d \n", r->info);
inorder(r->right);
}
}
void preorder(struct node *r)
{
if (r != NULL)
{
printf("%d \n", r->info);
preorder(r->left);
preorder(r->right);
}
}
void postorder(struct node *r)
{
if (r != NULL)
{
postorder(r->left);
postorder(r->right);
printf("%d \n", r->info);
}
}
void insert(int item)
{
struct node *temp,*q,*p;
temp= (struct node *)malloc(sizeof(struct node));
temp->info = item;
```

```
temp->left = temp->right = NULL;
if (root == NULL)
{
root=temp;
return;
}
q=root;
while(q!=NULL)
{
if(item==q->info)
{
printf("\nelement already exist");
return;
}
if(item<q->info)
{
p=q;
q=q->left;
}
else
if(item>q->info)
{
p=q;
q=q->right;
}
}
if(item>p->info)
{
p->right=temp;
return;
}
else
p->left=temp;
}
int main()
{
int item,ch;
while(1)
{
printf("\n1. insert node");
printf("\n2. display in inorder");
printf("\n3. display in preorder");
printf("\n4. display in postorder");
printf("\n5. exit");
printf("\nenter your choice:");
scanf("%d",&ch);
switch(ch)
{
case 1: printf("enter the element:");
scanf("%d",&item);
insert(item);
break;
case 2: inorder(root);
break;
case 3: preorder(root);
break;
case 4: postorder(root);
break;
case 5: printf("Program Ended\n");
break;
default:printf("Wrong Choice\n");
break;
}
}
return 0;
}
```

Output:

```
1. insert node
2. display in inorder
3. display in preorder
4. display in postorder
5. exit
enter your choice:1
enter the element:10

1. insert node
2. display in inorder
3. display in preorder
4. display in postorder
5. exit
enter your choice:1
enter the element:5

1. insert node
2. display in inorder
3. display in preorder
4. display in postorder
5. exit
enter your choice:2
5
10

1. insert node
2. display in inorder
3. display in preorder
4. display in postorder
5. exit
enter your choice: 5
```

A+

19/11/19

19/11/19

PROGRAM - 17

OBJECT: Program to implement BFS

Source Code:

```
#include<stdio.h>
int a[25][25];
int q[25];
int visited[25];
int  n;
int front = 0, rear = -1;
void bfs(int s) {
int i;
printf("%d",s);
for(i = 0; i < n; i++)
if(a[s][i]==1 && visited[i]==0)
q[++rear] = i;
if(front <= rear) {
visited[q[front]] = 1;
bfs(q[front++]);
}
}
int main() {
int s,i,j;
printf("\n Enter the number of vertices:");
scanf("%d", &n);
for(i=0; i < n; i++) {
q[i] = 0;
visited[i] = 0;
}
printf("\n Enter adjacency matrix:\n");
for(i=0; i<n; i++) {
for(j=0;j<n;j++) {
scanf("%d", &a[i][j]);
}
}
printf("\n Enter the source vertex:");
scanf("%d", &s);
bfs(s);
}
```

Dr. Somesh Kumar
Prof. & Head, CSE
Moradabad Institute of Technology
Moradabad-244001

19/11/19

PROGRAM - 17

Output:

```
Enter the number of vertices: 9
Enter Adjacency matrix:
0 1 1 1 0 0 0 0 0
0 0 0 0 1 0 0 0 0
0 1 0 0 0 0 0 0 0
0 0 1 0 0 0 1 0 0
0 0 1 0 0 0 0 0 0
0 0 1 0 1 0 0 1 0
0 0 1 0 0 1 0 0 0
0 0 0 0 1 0 0 0 1
0 0 0 0 0 1 1 0 0

Enter the source vertex: 0
0 1 2 3 4 6 5 7 8
```

27

19/11/19

Dr. Somesh Kumar
Prof. & Head, CSE
Moradabad Institute of Technology
Moradabad-244001

19/11/19

PROGRAM - 18

OBJECT: Program to implement DFS

Source Code:

```
#include<stdio.h>
#include<stdlib.h>
int G[25][25],visited[25],n;
 void dfs(int s,int d)
{
int j;
printf("\n%d",s);
visited[s]=1;
if(s==d)
{
printf("\ncompleted");
exit(0);
}
for(j=0;j<n;j++)
if(G[s][j]==1 && visited[j]==0)
dfs(j,d);
}
int main()
{
int i,j,s,d;
printf("Enter number of vertices:");
scanf("%d",&n);
printf("\nEnter adjecency matrix of the graph:");
for(i=0;i<n;i++)
for(j=0;j<n;j++)
scanf("%d",&G[i][j]);
for(i=0;i<n;i++)
visited[i]=0;
printf("\n enter the source node: ");
scanf("%d",&s);
printf("\n enter the destination: ");
scanf("%d",&d);
dfs(s,d);
}
```

Dr. Somesh Kumar
Prof. & Head, CSE
Moradabad Institute of Technology
Moradabad-244001

Output:

```
0 1 1 1 0 0 0 0 0
0 0 0 0 1 0 0 0 0
0 1 0 0 0 0 0 0 0
0 0 1 0 0 0 1 0 0
0 0 1 0 0 0 0 0 0
0 0 1 0 1 0 0 1 0
0 0 1 0 0 1 0 0 0
0 0 0 0 1 0 0 0 1
0 0 0 0 0 1 1 0 0

enter the source node: 8
enter the destination: 0

8
5
2
1
4
7
6
```

A

27/11/19

19/11/19 PROGRAM - 19

OBJECT: Program to implement Kruskal's algorithm to find minimum spanning tree of a given graph

Source Code:

```
#include<stdio.h>
#define MAX 30
typedef struct edge
{
int u,v,w;
}edge;
typedef struct edgelist
{
edge data[MAX];
int n;
}edgelist;
edgelist elist;
int adj[MAX][MAX],n;
edgelist spanlist;
void kruskal();
int find(int belongs[],int vertexno);
void union1(int belongs[],int t1,int t2);
void sort();
void print();
int main()
{
int i,j,total_cost;
printf("\nEnter number of vertices: ");
scanf("%d",&n);
printf("\nEnter the adjacency matrix:\n");
for(i=0;i<n;i++)
for(j=0;j<n;j++)
scanf("%d",&adj[i][j]);
kruskal();
print();
}
void kruskal()
{
int belongs[MAX],i,j,s1,s2;
elist.n=0;
for(i=1;i<n;i++)
for(j=0;j<i;j++)
{
if(adj[i][j]!=0)
{
elist.data[elist.n].u=i;
elist.data[elist.n].v=j;
elist.data[elist.n].w=adj[i][j];
elist.n++;
}
}
sort();
for(i=0;i<n;i++)
```

Dr. Somesh Kumar
Prof. & Head, CSE
Moradabad Institute of Technology
Moradabad-244001

```
belongs[i]=i;
spanlist.n=0;
for(i=0;i<elist.n;i++)
{
s1=find(belongs,elist.data[i].u);
s2=find(belongs,elist.data[i].v);
if(s1!=s2)
{
spanlist.data[spanlist.n]=elist.data[i];
spanlist.n=spanlist.n+1;
union1(belongs,s1,s2);
}
}
}
int find(int belongs[],int vertexno)
{
return(belongs[vertexno]);
}
void union1(int belongs[],int t1,int t2)
{
int i;
for(i=0;i<n;i++)
if(belongs[i]==t2)
belongs[i]=t1;
}
void sort()
{
int i,j;
edge temp;
for(i=1;i<elist.n;i++)
for(j=0;j<elist.n-1;j++)
if(elist.data[j].w>elist.data[j+1].w)
{
temp=elist.data[j];
elist.data[j]=elist.data[j+1];
elist.data[j+1]=temp;
}
}
void print()
{
int i,cost=0;
for(i=0;i<spanlist.n;i++)
{
printf("\n%d\t%d\t%d",spanlist.data[i].u,spanlist.data[i].v,spanlist.data[i].w);
cost=cost+spanlist.data[i].w;
}
printf("\n\nCost of the spanning tree=%d",cost);
}
```

Dr. Somesh Kumar
Prof. & Head, CSE
Moradabad Institute of Technology
Moradabad-244001

19/11/19

PROGRAM - 19

Output:

```
Enter number of vertices: 6

Enter the adjacency matrix:
0 3 1 6 0 0
3 0 5 0 3 0
1 5 0 5 6 4
6 0 5 0 0 2
0 3 6 0 0 6
0 0 4 2 6 0

2        0        1
5        3        2
1        0        3
4        1        3
5        2        4

Cost of the spanning tree=13
```

27

27/11/19

19/11/19 PROGRAM - 20

OBJECT: Program to implement warshall's algorithm to find all pair shortest path of a graph

Source Code:

```
#include<stdio.h>
#include<math.h>
int q[25][25];
int n;
int min(int a,int b)
{
if(a<b)
return(a);
else
return(b);
}
void warshal()
{
int i,j,k;
for(i=0;i<n;i++)
for(j=0;j<n;j++)
if(q[i][j]==0)
q[i][j]=999;
for(k=0;k<n;k++)
for(i=0;i<n;i++)
for(j=0;j<n;j++)
{
q[i][j]=min(q[i][j],q[i][k]+q[k][j]);
}
}
int main()
{
int i,j;
printf("\nEnter the number of vertices:");
scanf("%d",&n);
printf("\nEnter weight matrix of the graph:\n");
for(i=0;i<n;i++)
for(j=0;j<n;j++)
scanf("%d",&q[i][j]);
warshal();
printf("\n Transitive closure: \n");
for(i=0;i<n;i++)
{
for(j=0;j<n;j++)
printf("%d\t",q[i][j]);
printf("\n");
}
}
```

Dr. Somesh Kumar
Prof. & Head, CSE
Moradabad Institute of Technology
Moradabad-244001

Output:

```
Enter the number of vertices:6

Enter weight matrix of the graph:
0 3 1 6 0 0
3 0 5 0 3 0
1 5 0 5 6 4
6 0 5 0 0 2
0 3 6 0 0 6
0 0 4 2 6 0

Transitive closure:
2      3      1      6      6      5
3      6      4      9      3      8
1      4      2      5      6      4
6      9      5      4      8      2
6      3      6      8      6      6
5      8      4      2      6      4
```

A+

27/11/19