MES KALLADI COLLEGE MANNARKKAD

DEPARTMENT OF COMPUTER SCIENCE

PROGRAMMING LAB: DATA STRUCTURE USING C

IVth Semester BCA & B.Sc COMPUTER SCIENCE

INDEX

PROGRAM:1-Reverse a String using Pointer

Source Code:

```
#include<stdio.h>

#include<conio.h>

int main()

{

char str[50];

char revstr[50];

char *strptr=str;

char *revptr=revstr;

int len=-1;

clrscr();

printf("enter the string:\n");

scanf("%s",str);

while(*strptr)

{

strptr++;

len++;

}

while(len>=0)

{

strptr--;

*revptr=*strptr;
revptr++;

--len;

}

*revptr='\0';

printf("reverse of a string is \n\n%s",revstr);

getch();

return 0;

}
```

PROGRAM:2 –Implement Pattern Matching Algorithm

Source Code:

```
#include<stdio.h>

#include<string.h>

#include<conio.h>

void main()

{

char txt[20],pat[20];

int a,b,i,j;

clrscr();

printf("enter the string:\n");

gets(txt);

 printf("enter the patern to find:\n");

gets(pat);


a=strlen(pat);
b=strlen(txt);
for(i=0;i<=b-a;i++)
{
for(j=0;j<a;j++)
if(txt[i+j]!=pat[j])
break;
if(j==a)
printf("\n pattern found at index %d\n",i+1);
}
getch();
}
```

PROGRAM:3 –Search an element in the 2-dimensional Array

Source Code:

```
#include<stdio.h>
#include<conio.h>
void main()
{
int m,n,i,j,srchno,count=0,a[50][50];
clrscr();
printf("Enter number of rows and column:\n");
scanf("%d%d",&m,&n);

printf("Enter %d elements :\n",(m*n));
for(i=0;i<m;i++)
{
for(j=0;j<n;j++)
{
scanf("%d",&a[i][j]);
}
}
printf("Enter elements to get the position:\t");
scanf("%d",&srchno);
for(i=0;i<m;i++)
{
for(j=0;j<n;j++)
{
if(a[i][j]==srchno)
{
printf("(%d %d)\n",i,j);
count++;
}
}
}
if(count==0)
printf("not Found");
getch();
}
```

PROGRAM:4 –Append two Arrays

Source Code:

```
#include<stdio.h>

#include<conio.h>

void main()

{

int ar[30],br[30],cr[30],i,j,m,n;

clrscr();

printf("\n enter limit of 1st array:");

scanf("%d",&m);

printf("\n enter limit of 2nd array:");

scanf("%d",&n);

printf("\n enter elements  of 1st array:");

for(i=0;i<m;i++)

{

scanf("%d",&ar[i]);

  }

printf("\n enter elements of 2nd array:");

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

{

scanf("%d",&br[j]);

  }

  for(i=0;i<m;i++)

  cr[i]=ar[i];

  for(j=0;j<n;j++)
  cr[i+j]=br[j];
  printf("\n after appending array is:");
  for(i=0;i<m+n;i++)
  {
  printf("%d\t",cr[i]);
  }
  getch();
  }
```

PROGRAM:5 -Search an element in the array using binary Search

Source Code:

```
#include<stdio.h>
#include<conio.h>
void main()
{
int list[25],max,first,last,middle,i,item,loc=-1;
clrscr();
printf("\n enter the limit:");
scanf("%d",&max);
printf("\n enter  array elements:");
for(i=0;i<max;i++)
{
scanf("%d",&list[i]);
}
printf("\n Enter item to be searched:");
scanf("%d",&item);
first=0;

last=max-1;

while(first<=last)

{

middle=(first+last)/2;

if(item==list[middle])

{

loc=middle;

break;

}

if(item<list[middle])

last=middle-1;

else

first=middle+1;

}

if(loc!=-1)

printf("\n the item is found at position %d",loc+1);

else

printf("not found");

getch();

}
```

PROGRAM:6-Read a sparse matrix and display its triplet representation using array

Source Code:

```
#include<stdio.h>

#include<conio.h>
```

```
void main()
{
int i,j,m,n,ar[10][10],br[10][10],s=0;
clrscr();
printf("\n enter order of matrix :");
scanf("%d%d",&m,&n);
printf("\n  elements of matrix:");
for(i=0;i<m;i++)
{
for(j=0;j<n;j++)
{
scanf("%d",&ar[i][j]);
}
}
printf("\n the given matrix is:\n");
 for(i=0;i<m;i++)
{
for(j=0;j<n;j++)
{
printf("%d\t",ar[i][j]);
}
printf("\n");
}
for(i=0;i<m;i++)
{
```

```
for(j=0;j<n;j++)
{
if(ar[i][j]!=0)
{
br[s][0]=i;
br[s][1]=j;
br[s][2]=ar[i][j];
s++;
}
}
}
printf("the sparse matrix is:\n ");
for(i=0;i<s;i++)
{
for(j=0;j<3;j++)
{
printf("%d\t",br[i][j]);
}
printf("\n");
}
getch();
}
```

PROGRAM:7-Create a singly linked list of n nodes and display it

Source Code:

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

```
#include<conio.h>

#include<stdlib.h>

struct node

{

int data;

struct node *nextptr;

};

struct node*stNode;

static void createList(int n);

static void displaylist();


static void createList(int n)

{

struct node *nNode;

struct node *ndBuffer;

int nData;

int i;

stNode=(struct node*)malloc(sizeof(struct node));

if(stNode==NULL)

{

printf("memory can not be allocated");

}

else

{

printf("Input data for node 1:");
scanf("%d",&nData);

stNode->data=nData;

stNode->nextptr=NULL;

ndBuffer=stNode;

for(i=2;i<=n;i++)

{

nNode=(struct node *)malloc(sizeof(struct node));

if(nNode==NULL)

{

printf("memory can not be allocated");

break;

}

else

{

printf("input data for node %d :",i);

scanf("%d",&nData);

nNode->data=nData;

nNode->nextptr=NULL;

ndBuffer->nextptr=nNode;

ndBuffer=ndBuffer->nextptr;

}

 }

 }

 }

 static void displaylist()
 {
 struct node *ndBuffer;
 ndBuffer=stNode;
 if(ndBuffer==NULL)
 {
 printf("list is empty");
 }
 else
 {
 while(ndBuffer!=NULL)
 {
 printf("Data=%d\n",ndBuffer->data);
 ndBuffer=ndBuffer->nextptr;
 }
 }
 }
void main()
{
int  num;
clrscr();
printf("Input the number of nodes:");
scanf("%d",&num);
createList(num);
printf("Data entered in the list\n");
displaylist();

getch();

}
```

PROGRAM: 8 –Delete a given node from a singly linked list

Source Code:

```
#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

struct node

{

int num;

struct node *nextptr;

}*stnode;

void create(int n);

void delete(int pos);

void display();


 void create(int n)

 {

 struct node  *fnnode,*tmp;

 int num,i;

 stnode=(struct node *)malloc(sizeof(struct node));

 if(stnode==NULL)

 {

 printf("memory can not be allocated");
```

```
}
else
{
printf("Input data for node 1 \n");
scanf("%d",&num);
stnode->num=num;
stnode->nextptr=NULL;
tmp=stnode;
for(i=2;i<=n;i++)
{
fnnode=(struct node *)malloc(sizeof(struct node));
if(fnnode==NULL)
{
printf("memory can not be allocated");
break;
}
else
{
printf("Input data for node %d \n",i);
scanf("%d",&num);
fnnode->num=num;
fnnode->nextptr=NULL;
tmp->nextptr=fnnode;
tmp=tmp->nextptr;
}
```

```
 }

 }
}
void delete(int pos)
{
int  i;
struct node *todel,*prenode;
if(stnode==NULL)
{
printf("There is no nodes in the list");
}
else
{
todel=stnode;
prenode=stnode;
for(i=2;i<=pos;i++)
{
prenode=todel;
todel=todel->nextptr;
if(todel==NULL)
break;
}
if(todel!=NULL)
{
```

```
if(todel==stnode)

stnode=stnode->nextptr;

prenode->nextptr=todel->nextptr;

todel->nextptr=NULL;

free(todel);

}

else

{

printf("Deletion can not be possible from that position");

}

}

}

void display()

{

 struct node *tmp;

 if(stnode==NULL)

 {

 printf("No data found in the list");

 }

 else

 {

 tmp=stnode;

 while(tmp!=NULL)

 {

 printf("Data=%d\n",tmp->num);
 tmp=tmp->nextptr;
 }
 }
 }
void main()
{
int n,num,pos;
clrscr();
printf("Input the number of nodes:\t");
scanf("%d",&n);
create(n);
printf("Data entered in the list are:\n");
display();

printf("\nInput the position of node to delete:\t");
scanf("%d",&pos);

if(pos<=1||pos>=n)
{
printf("Deletion can not be possible from that position\n");
}
  if(pos>1 && pos<n)
  {
  printf("Deletion  completed successfully\n");
     delete(pos);
```

```
}

printf("The new list are:\n");

display();

getch();

}
```

PROGRAM:9 –Create a doubly linked list of integers and display in forward and backward directions

Source Code:

```
#include<stdio.h>

#include<stdlib.h>

#include<conio.h>

struct node

{

int data;

struct node *rptr,*lptr;

};

 struct node * create(struct node *,struct node **,int);

 void display(struct node *);

 void displays(struct node *,struct node *);


struct node*create(struct node *head1,struct node **tail1,int dat)

{

  struct node *newnode,*temp;

  newnode=(struct node*)malloc(sizeof(struct node));

  newnode->data=dat;
  newnode->rptr=newnode->lptr=NULL;
  if(head1==NULL)
   {
     newnode->lptr=newnode->rptr=NULL;
     head1=newnode;
   }
 temp=head1;
 while(temp->rptr!=NULL)
 temp=temp->rptr;
 temp->rptr=newnode;
 newnode->lptr=temp;
 newnode->rptr=NULL;
 *tail1=newnode;
 temp=temp->rptr;
 return head1;
}

void display(struct node *head)
{
while(head!=NULL)
{
printf("%d\n",head->data);
head=head->rptr;
}
}
```

```
void displays(struct node *tail,struct node *head)
{
while(tail!=head)
{
printf("%d\n",tail->data);
tail=tail->lptr;
}

if(tail==head)
printf("%d\n",tail->data);
}

void main()
{

  int i,n,value;
  struct node *head,*tail;
  head=NULL;
  tail=NULL;
  clrscr();
  printf("Enter the   limit:");
  scanf("%d",&n);
  for(i=0;i<n;i++)
  {
```

```
 printf("Enter the numbers:");

 scanf("%d",&value);

 head=create(head,&tail,value);

 }

 printf("\nThe data in the forward direction is printed below\n");

 display(head);

 printf("\nThe data in the backward direction is  printed below\n");

 displays(tail,head);

 getch();

 }
```

PROGRAM:10 –Implement stack operation using array

Source Code:

```
#include<stdio.h>

#include<conio.h>

int stack[100],choice,top,n,val,i;


void push();

void pop();

void display();


int main()

{


top=-1;

clrscr();
printf("enter size of the stack:\t");
scanf("%d",&n);
printf("--------stack operation-------");
printf("\n 1.push 2.pop 3.display 4.exit\n");
do
{
printf("\n enter your choice:");
scanf("%d",&choice);
switch(choice)
{
case 1:push();
     break;
case 2: pop();
        break;
case 3: display();
        break;
case 4:printf("exit point");
     break;
default:printf("invalid choice");
}
}while(choice!=4);
return 0;
}
```

```
void push()

{

if(top<n-1)

{

printf("\n  enter elements to be pushed:");

scanf("%d",&val);

top++;

stack[top]=val;

}

else

{

printf("stack overflow");

}

}


void pop()

{

if(top>-1)

{

printf("the popped elements is %d",stack[top]);

top--;

}

else

{

printf("stack underflow");
}

}


void display()

{

if(top>=0)

{

printf("the elements of stack are:\n");

for(i=top;i>=0;i--)

printf("%d\n",stack[i]);

}

else

{

printf("stack is empty");

}

}
```

PROGRAM:11 -Stack Operation using Linked List

Source Code:

```
#include<stdio.h>

#include<conio.h>

 struct node

 {

 int info;

 struct node *ptr;
 }*top,*top1,*temp;

int push(int a);
void pop() ;
void display();
int count=0;

int main()
{
  int choice,val;

clrscr();
printf("--------stack operation-------");
printf("\n 1.push 2.pop 3.display 4.exit\n");
do
{
printf("\n enter your choice:");
scanf("%d",&choice);
switch(choice)
{
case 1:printf("enter elements to be pushed: ");
       scanf("%d",&val);
       push(val);
       break;
case 2: pop();
            break;
case 3: display();
            break;
case 4:printf("exit point");
       break;
default:printf("invalid choice");
}
}while(choice!=4);
return 0;
}
int push(int a)
{
if(top==NULL)
{
  top=(struct node*)malloc(1*sizeof(struct node));
  top->ptr=NULL;
  top->info=a;
  }
  else
  {
  temp=(struct node *)malloc(1*sizeof(struct node));
  temp->info=a;
  temp->ptr=top;
  top=temp;
  }
  count++;
  return 0;
  }
void pop()
{
top1=top;
if(top1==NULL)
{
printf("stack underflow");
}
else
{
top1=top1->ptr;
printf("the popped elements is %d\n",top->info);
free(top);
top=top1;
count--;
}
}
void display()
{
top1=top;
if(top1==NULL)
{
 printf("stack is empty");
}
else
{
printf("the elements  are:\n");
while(top1!=NULL)
{
printf("%d\n",top1->info);
top1=top1->ptr;
}
}
}
```

PROGRAM:12-Evaluation of postfix expression

Source Code:

```
#include<stdio.h>
#include<conio.h>
int stack[20];
int top=-1;
void push(int x)
{
stack[++top]=x;
}
int pop()
{
return stack[top--];
```

```
}
void main()
{
char exp[20];
char *e;
int n1,n2,n3,num;
clrscr();
printf("enter the postfix Expression:\t");
scanf("%s",exp);
e=exp;
while(*e!='\0')
{
if(isdigit(*e))
{
num=*e-48;
push(num);
}
else
{
n1=pop();
n2=pop();
switch(*e)
{
case'+':
{
```

```
n3=n1+n2;

break;

}

case'-':

{

n3=n2-n1;

break;

}

case'*':

{

n3=n1*n2;

break;

}

case'/':

{

n3=n2/n1;

break;

}

}

push(n3);

}

e++;

}

printf("The result of the postfix expression %s=%d\n\n",exp,pop());

getch();
```

```
}
```

PROGRAM:13-Implement Queue using Array

Source Code:

```
#include<stdio.h>
#include<conio.h>
int array[100],n,front=-1,rear=-1,val,choice,i;

void insert();
void delete();
void display();

int main()
{
clrscr();
printf("enter size of the queue:\t");
scanf("%d",&n);
printf("\n--------Queue operation-------\n");
printf("\n 1.insert 2.delete 3.display 4.exit\n");
do
{
printf("\n enter your choice:");
scanf("%d",&choice);
switch(choice)
{
case 1:insert();
```

```
       break;
case 2: delete();
            break;
case 3: display();
            break;
case 4:printf("exit point");
       break;
default:printf("invalid choice");
}
}while(choice!=4);
return 0;
}
void insert()
{
if(rear==n-1)

printf("overflow");
else

{
if(front==-1)
front=0;
printf("\n  enter elements to be inserted:");
scanf("%d",&val);
rear=rear+1;
```

```
array[rear]=val;

}
}
void delete()
{
if(front==-1)
{
printf("Queue underflow");
return;
}
else
{
printf("the deleted elements is %d",array[front]);
front=front+1;
}
}

void display()
{
if(front==-1)
printf("Queue is empty");

else
{
```

```
printf("the elements of queue are:\n");
for(i=front;i<=rear;i++)
printf("%d\n",array[i]);
}
}
```

PROGRAM:14- Implement Queue using Linked List

Source Code:

```
#include<conio.h>
#include<stdio.h>
#include<stdlib.h>
struct node
{
int info;
struct node *next;
};
typedef struct node *link;
link q;

link getnode()
{
link q;
q=(link)malloc(sizeof(struct node));
return(q);
}
```

```
 void insert(link s,int y)
  {
  link p;
  p=getnode();
  p->info=y;
  p->next=NULL;
  if(s->next==NULL)
  s->next=p;
  else
  q->next=p;
  q=p;

  }
void display(link s)
{
  link p;
  p=getnode();
  p=s->next;
  while(p!=NULL)
  {
  printf("%d\t",p->info);
  p=p->next;
  }

   }
   void freenode(link p)

   {

   free(p);

   }


  int delete(link s)

  {

  link p;

  int y;

  p=getnode();

  p=s->next;

  if(p==q)

  q=s;

  else

  s->next=p->next;

  y=p->info;

  freenode(p);

  return(y);


}

void main()

{

    link s;

    int x,y;

    clrscr();
```

```
     s=getnode();
     q=s;
    printf("\n\n 1.insert \n 2.delete the data \n 3.display the data \n 4.exit\n");
     do
     {

   printf("\nenter your choice:");
   scanf("%d",&x);

     switch(x)
{
case 1:
             printf("Enter the number to insert:");
             scanf("%d",&y);
             insert(s,y);
              break;

case 2:   if(q!=s)
              {
              y=delete(s);
              printf("deleted number is %d:",y);
              }
              else
              printf("underflow");
              break;
case 3:
            display(s);
            break;
}}
while(x<4);
 getch();
 }
```

PROGRAM:15-Search an element in a binary search tree

Source Code:

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,first,last,middle,n,search,array[100];
clrscr();
printf("\n enter number of elements :");
scanf("%d",&n);
printf("\n enter  %d integers\n",n);
for(i=0;i<n;i++)
scanf("%d",&array[i]);
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 is not present in the list\n");

 getch();
             }
```

PROGRAM:16- Implement Exchange Sort

Source Code:

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

```
void main()
{
int a[100],i,n,j,t;
clrscr();
printf("enter a limit:\t");
scanf("%d",&n);
printf("Enter the elements to be sorted:\n");
for(i=0;i<n;i++)
scanf("%d",&a[i]);
for(i=0;i<n-1;i++)
for(j=i+1;j<n;j++)
if(a[i]>a[j])
{
t=a[i];
a[i]=a[j];
a[j]=t;
}
printf("the sorted list of elements:\n");
for(i=0;i<n;i++)
printf("%d\n",a[i]);
getch();
}
```

PROGRAM:17

Selection Sort

Source Code:

```
#include<stdio.h>

#include<conio.h>

void main()

{

int ar[25],n,i,j,min,pos;

clrscr();

printf("Enter the limit:\n");

scanf("%d",&n);

printf("Enter elements:\n");

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

scanf("%d",&ar[i]);

for(i=0;i<n-1;i++)

{

min=ar[i];

pos=i;

for(j=i+1;j<n;j++)

{

if(ar[j]<min)

{

min=ar[j];

pos=j;

}

}

if(pos!=i)

{
```

```
ar[pos]=ar[i];
ar[i]=min;
}
}
printf("sorted array is \n");
for(i=0;i<n;i++)
{
printf("%d\t",ar[i]);
}
getch();
}
```

PROGRAM:18- Implement Insertion Sort

Source Code:

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a[100],i,n,j,temp;
clrscr();
printf("enter no of elements:");
scanf("%d",&n);
printf("Enter the elements:");
for(i=0;i<n;i++)
{
scanf("%d",&a[i]);
}
for(i=1;i<=n-1;i++)
{
j=i;
while(j>0 && a[j-1]>a[j])
{
temp=a[j];
a[j]=a[j-1];
a[j-1]=temp;
j--;
}
}
printf("The sorted  elements are:\n");
for(i=0;i<=n-1;i++)
{
printf("%d\n",a[i]);
}
getch();
}
```

******************************END***************************************