DSC LAB PROGRAMS

1.Design, Develop and Implement a menu-driven Program in C for the following Array operations

Creating an Array of N Integer Elements

Display of Array Elements with Suitable Headings

Inserting an Element (ELEM) at a given valid Position (POS)

Deleting an Element at a given valid Position(POS)

Exit.

Support the program with functions for each of the above operations.

#include<stdio.h>

#include<stdlib.h>

int a[10],n,elem,i,pos;

void create();

void display();

void insert();

void delete();

void create()

{

int i;

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]);

}

void display()

{

int i;

printf("The array elements are:\n");

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

printf("%d\t",a[i]);

}

void insert()

{

int i;

printf("Enter the position for new element:\n");

scanf("%d",&pos);

printf("Enter the element to be inserted:\n");

scanf("%d",&elem);

for (i = n-1; i <=pos; i--)

a[i+1]=a[i];

a[pos]=elem;

n=n+1;

}

void delete()

{

printf("Enter position of element to be deleted:\n");

scanf("%d",&pos);

elem=a[pos];

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

a[i]=a[i+1];

n=n-1;

printf("Deleted element is %d\n",elem );

}

int main()

{

int ch;

while(ch)

{

printf("\n\n\_\_\_\_\_\_MENU\_\_\_\_\_\n");

printf("1.Create\n2.Display\n3.Insert\n4.Delete\n5.Exit\n");

printf("Enter Your Choice:");

scanf("%d",&ch);

switch(ch)

{

case 1:create();break;

case 2:display();break;

case 3:insert();break;

case 4:delete();break;

case 5:exit(0);break;

default :printf("INVALID CHOICE\n");

}

}return 0;

}

2. Write a C program to create a sequential file with at least five records. Each record having the structure show below:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Name | Marks1 | Marks2 | Marks3 |
| Non-Zero  Positive | 25-Character | Positive Integer | Positive Integer | Positive Integer |

1. To display all the records in the file
2. To search for a specific record based on the USN. In case the record is not found. Suitable message should be displayed. Both the options in this case must be demonstrated.

Program:

#include<stdio.h>

#include<conio.h>

#include<string.h>

struct student

{

char name[10];

int usn;

int marks1,marks2,marks3;

};

main()

{

int n,avg[10],i,j;

struct student a[10];

clrscr();

printf("enter the number of student:\n");

scanf("%d",&n);

printf("enter the details of student\n");

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

{

printf("name=");

scanf("%s",a[i].name);

printf("usn=");

scanf("%d",&a[i].usn);

printf("marks1=");

scanf("%d",&a[i].marks1);

printf("marks2=");

scanf("%d",&a[i].marks2);

printf("marks3=");

scanf("%d",&a[i].marks3);

avg[i]=(a[i].marks1+a[i].marks2+a[i].marks3)/3;

}

printf("the details of students is:\n");

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

{

printf("name\t usn\t marks1\t marks2\t marks3\t average\n");

printf("%s\t %d\t %d\t %d\t %d\t %d\n",a[i].name,a[i].usn,a[i].marks1,a[i].marks2,a[i].marks3,avg[i]);

getch();

}

return;

}

**3.** Design a C program to show the usage of Dynamic memory allocation techniques

Program to calculate the sum of n numbers entered by the user

#include <stdio.h>

#include <stdlib.h>

int main() {

int n, i, \*ptr, sum = 0;

printf("Enter number of elements: ");

scanf("%d", &n);

ptr = (int\*) malloc(n \* sizeof(int));

// if memory cannot be allocated

if(ptr == NULL) {

printf("Error! memory not allocated.");

exit(0);

}

printf("Enter elements: ");

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

scanf("%d", ptr + i);

sum += \*(ptr + i);

}

printf("Sum = %d", sum);

// deallocating the memory

free(ptr);

return 0;

}

4. Design, Develop and Implement a menu driven Program in C for the following operations on STACK of Integers (Array Implementation of Stack with maximum size MAX).

Push an Element on to Stack

Pop an Element from Stack

Demonstrate Overflow and Underflow situations on Stack

Display the status of Stack

Exit

#include<stdio.h>#include<conio.h>int i,top,ch,s[3],item;void push(),pop(),dis();

main(){clrscr();top=-1;ch=0;while(ch!=4){ printf("\nEnter the menu for stack operation\n"); printf("\n1:insert\n2:delete\n3:display\n4:exit\n"); printf("Input ur choice\n"); scanf("%d",&ch); switch(ch) { case 1: push(); break; case 2: pop(); break; case 3: dis(); break; case 4: exit(0); break; default: printf("Invalid choice\n"); } }getch();return 0;}

void push() { int item; if(top==3-1) { printf("Stack Overflow\n"); return; } printf("Enter an item to be pushed:"); scanf("%d",&item); top+=1; s[top]=item; }

void pop() { if(top==-1) { printf("Stack Undeflow\n"); return; } printf("Item poped is %d\n",s[top--]); }

void dis() { if(top==-1) { printf("Stack is empty\n"); return; } printf("\nStack contains......\n"); for(i=0; i<=top; i++) printf("%d\t",s[i]); }

* 5.Design, Develop and Implement a Program in C for converting an Infix Expression to Postfix Expression. Program should support for both parenthesizedand free parenthesized expressions with the operators: +, -, \*, /, %(Remainder), ^(Power) and alphanumeric operands.

Program:

#include<stdio.h>

#include<conio.h>

#include<string.h>

int stack[50],top=-1;

char postfix[25];

int main(void)

{

char infix[25];

clrscr();

printf("enter a valid infix expression:");

gets(infix);

infix\_to\_postfix(infix);

getch();

return 0;

}

infix\_to\_postfix(char infix[])

{

int length;

static int index=0,pos=0;

char symbol,temp,postfix[50];

length=strlen(infix);

push('#');

while(index<length)

{

symbol=infix[index];

switch(symbol)

{

case'(':push(symbol);

break;

case')':temp=pop();

while(temp!='(')

{

postfix[pos]=temp;

pos++;

temp=pop();

}

break;

case'+':

case'-':

case'\*':

case'/':while(preced(stack[top])>=preced(symbol))

{

temp=pop();

postfix[pos]=temp;

pos++;

}

push(symbol);

break;

default:postfix[pos++]=symbol;

break;

}

index++;

}

while(top>0)

{

temp=pop();

postfix[pos++]=temp;

}

postfix[pos++]='\0';

printf("the postfix of %s is=%s",infix,postfix);

return;

}

int preced(char ch)

{

switch(ch)

{

case'\*':

case'/':return 2;

case'+':

case'-':return 1;

case'(':

case')':return 0;

case'#':return -1;

}

return 0;

}

push(char ch)

{

top++;

stack[top]=ch;

return;

}

pop()

{

char ch;

ch=stack[top];

top--;

return ch;

}

* 6.Design, Develop and Implement a Program in C for the following Stack Applications
  1. Evaluation of Suffix expression with single-digit operands and operators:+, -, \*, /, %, ^
  2. Solving Tower of Hanoi problem with n disks.

Program:

#include<stdio.h>

#include<math.h>

#include<string.h>

#include<ctype.h>

double compute(char symbol,double op1,double op2)

{

switch(symbol)

{

case'+':return op1+op2;

case'-':return op1-op2;

case'\*':return op1\*op2;

case'/':return op1/op2;

case'$':

case'^':return pow(op1,op2);

}

return 0;

}

void main()

{

double s[20];

double res;

double op1;

double op2;

int top;

int i;

char postfix[20];

char symbol;

printf("Enter the postfix expression\n");

scanf("%s",postfix);

top=-1;

for(i=0;i<strlen(postfix);i++)

{

symbol=postfix[i];

if(isdigit(symbol))

s[++top]=symbol-'0';

else

{

op2=s[top--];

op1=s[top--];

res=compute(symbol,op1,op2);

s[++top]=res;

}

}

res=s[top--];

printf("The result is %f\n",res);

}

Tower of Hanoi

Program:

#include<stdio.h>

#include<conio.h>

main()

{

int n;

clrscr();

printf("Enter the number of discs\n");

scanf("%d",&n);

towers(n,'A','C','B');

getch();

return;

}

towers(int n,char scr,char dst,char temp)

{

if(n==1)

printf("Move disc %d from %c to %c\n",n,scr,dst);

else

{

towers(n-1,scr,temp,dst);

printf("Move disc %d from %c to %c\n",n,scr,dst);

towers(n-1,temp,dst,scr);

}

return;

}

* 7.Design, Develop and Implement a menu driven Program in C for the following operations on QUEUE of Characters (Array Implementation of Queue with maximum size MAX)
  1. Insert an Element on to QUEUE
  2. Delete an Element from QUEUE
  3. Demonstrate Overflow and Underflow situations on QUEUE
  4. Display the status of QUEUE
  5. Exit

Program:

#include<stdio.h>

#include<conio.h>

int queue[3],maxsize=3;

int front=-1,rear=-1;

main()

{

int ch;

clrscr();

do

{

printf("Enter the choice/n");

printf("1->Add\t2->Delete\t3->Display\t4->Exit\t");

scanf("%d",&ch);

switch(ch)

{

case 1:insert();

break;

case 2:delete();

break;

case 3:display();

break;

case 4:return;

default:printf("Wrong choice\n");

}

}

while(ch!=4);

return;

}

insert()

{

int num;

if(rear==(maxsize-1))

{

printf("Queue is full\n");

return;

}

else

{

printf("Enter the element:");

scanf("%d",&num);

rear++;

queue[rear]=num;

if(front==-1)

front=rear;

}

return;

}

delete()

{

if(front==-1)

{

printf("Queue is empty\n");

return;

}

else

{

printf("Deleted element is %d\n",queue[front]);

if(front==rear)

front=-1;

else

front++;

}

return;

}

display()

{

int i;

if(front==-1)

{

printf("Queue is empty\n");

return;

}

else

{

printf("The element are:");

for(i=front;i<=rear;i++)

printf("%d",queue[i]);

}

printf("\n");

return;

}

* 8.Design, Develop and Implement a menu driven Program in C for the following operations on Circular QUEUE of Characters (Array Implementation of Queue with maximum size MAX)
  1. Insert an Element on to Circular QUEUE
  2. Delete an Element from Circular QUEUE
  3. Demonstrate Overflow and Underflow situations on Circular QUEUE
  4. Display the status of Circular QUEUE
  5. Exit

Program:

#include<stdio.h>

#include<process.h>

#idefine queue\_size 5

void insert\_rear(int item,int q[],int \*r,int \*count)

{

if(\*count==queue\_size)

{

printf("overflow of queue\n");

return;

}

\*r=(\*r+1)%queue\_size;

q[\*r]=item;

\*count+=1;

}

void delete\_front(int q[],int \*f,int \*count)

{

if(count==0)

{

printf("underflow of queue\n");

return;

}

printf("the deleted element is %d/n",q[\*f]);

\*f=(\*f+1)%queue\_size;

\*count=1;

}

void display(int q[],int f,int count)

{

int i;

if(count==0)

{

printf("q is empty\n");

return;

}

printf("contents of queue is\n");

for(i=1;i<=count;i++)

{

printf("%d\n",q[f]);

f=(f+1)%queue\_size;

}

}

void main()

{

int choice,item,f,r,count,q[20];

f=0;

r=-1;

count=0;

clrscr();

for(;;)

{

printf("1:insert 2:delete\n");

printf("3:display 4:exit\n");

printf("enter the choice\n");

scanf("%d",&choice);

switch(choice);

{

case 1:printf("enter the item to be inserted\n");

scanf("%d",&item);

insert\_rear(item,q,&r,&count);

break;

case 2:delete\_front(q,&f,&count);

break;

case 3:display(q,f,count);

break;

default:exit(0);

}

}

}

9.Write a C program using dynamic variables and pointers, to construct a singly linked list consisting of the following information in each node: student id (interger), student name(character string) and semester(integer). The operations to be supposed are:

1. The insertion operation

i)At the front of a list

ii) At the back of the list

1. Deleting a node based on student id. If the specified node is not present in the list an error message should be displayed. Both the operation should be demonstrated.
2. Displaying all the nodes in the list.

Program:

#include<stdio.h>

#include<alloc.h>

#include<process.h>

#include<string.h>

struct student

{

char name[20];

int id;

int sem;

struct student \*link;

};

typedef struct student \*STUDENT;

STUDENT getnode(void)

{

STUDENT X;

X=(STUDENT)malloc(sizeof(struct student));

if(X==NULL)

{

printf("out of memory\n");

exit(0);

}

return X;

}

void freenode(STUDENT X)

{

free(X);

}

STUDENT insert\_front(char name[],int id,int sem,STUDENT first)

{

STUDENT temp;

temp=getnode();

strcpy(temp->name,name);

temp->sem=sem;

temp->id=id;

temp->link=first;

first=temp;

return first;

}

STUDENT insert\_rear(char name[],int id,int sem,STUDENT first)

{

STUDENT temp;

STUDENT cur;

temp=getnode();

strcpy(temp->name,name);

temp->id=id;

temp->sem=sem;

temp->link=NULL;

if(first==NULL)return temp;

cur=first;

while(cur->link!=NULL)

{

cur=cur->link;

}

cur->link=temp;

return first;

}

STUDENT delete\_student(int id,STUDENT first)

{

STUDENT pre,cur;

if(first==NULL)

{

printf("no student in the organization\n");

return NULL;

}

pre=NULL;

cur=first;

while(cur!=NULL && id!=cur->id)

{

pre=cur;

cur=cur->link;

}

if(cur==NULL)

{

printf("student id not found\n");

return first;

}

if(pre==NULL)

first=first->link;

else

pre->link=cur->link;

free(cur);

return first;

}

void display(STUDENT first)

{

STUDENT temp;

if(first==NULL)

{

printf("no student in the organization\n");

return;

}

printf("studentname studentid studentsem\n");

printf("................\n");

for(temp=first;temp!=NULL;temp=temp->link)

printf("%10s %4d %4d\n",temp->name,temp->id,temp->sem);

printf("\n");

}

void main()

{

STUDENT first=NULL;

int choice,id,sem,pos;

char name[10];

clrscr();

for(;;)

{

printf("1:insert front 2:insert rear\n");

printf("3:delete 4:display\n");

printf("5:exit\n");

printf("enter the ur choice\n");

scanf("%d",&choice);

if(choice==1 || choice==2)

{

printf("name:");

scanf("%s",name);

printf("id:");

scanf("%d",&id);

printf("sem:");

scanf("%d",&sem);

}

switch(choice)

{

case 1:

first=insert\_front(name,id,sem,first);

break;

case 2:

first=insert\_rear(name,id,sem,first);

break;

case 3:

printf("delete student details for id:");

scanf("%d",&id);

first=delete\_student(id,first);

break;

case 4:

display(first);

break;

default:

exit(0);

}

}

}

* 10.Design, Develop and Implement a menu driven Program in C for the following operations on Doubly Linked List (DLL) of Employee Data with the fields: SSN, Name, Dept, Designation, Sal, PhNo
  1. Create a DLL of N Employees Data by using end insertion.
  2. Display the status of DLL and count the number of nodes in it
  3. Perform Insertion and Deletion at End of DLL
  4. Perform Insertion and Deletion at Front of DLL
  5. Demonstrate how this DLL can be used as Double Ended Queue
  6. Exit

#include<string.h>

int count=0;

struct node

{

struct node \*prev;

int ssn,phno;

float sal;

char name[20],dept[10],desg[20];

struct node \*next;

}\*h,\*temp,\*temp1,\*temp2,\*temp4;

void create()

{

int ssn,phno;

float sal;

char name[20],dept[10],desg[20];

temp =(struct node \*)malloc(sizeof(struct node));

temp->prev = NULL;

temp->next = NULL;

printf("\n Enter ssn,name,department, designation, salary and phno of employee : ");

scanf("%d %s %s %s %f %d", &ssn, name,dept,desg,&sal, &phno);

temp->ssn = ssn;

strcpy(temp->name,name);

strcpy(temp->dept,dept);

strcpy(temp->desg,desg);

temp->sal = sal;

temp->phno = phno;

count++;

}

void insertbeg()

{

if (h == NULL)

{

create();

h = temp;

temp1 = h;

}

else

{

create();

temp->next = h;

h->prev = temp;

h = temp;

}

}

void insertend()

{

if(h==NULL)

{

create();

h = temp;

temp1 = h;

}

else

{

create();

temp1->next = temp;

temp->prev = temp1;

temp1 = temp;

}

}

void displaybeg()

{

temp2 =h;

if(temp2 == NULL)

{

printf("List empty to display \n");

return;

}

printf("\n Linked list elements from begining : \n");

while (temp2!= NULL)

{

printf("%d %s %s %s %f %d\n", temp2->ssn, temp2->name,temp2->dept,

temp2->desg,temp2->sal, temp2->phno );

temp2 = temp2->next;

}

printf(" No of employees = %d ", count);

}

int deleteend()

{

struct node \*temp;

temp=h;

if(temp->next==NULL)

{

free(temp);

h=NULL;

return 0;

}

else

{

temp2=temp1->prev;

temp2->next=NULL;

printf("%d %s %s %s %f %d\n", temp1->ssn, temp1->name,temp1->dept,

temp1->desg,temp1->sal, temp1->phno );

free(temp1);

}

count--;

return 0;

}

int deletebeg()

{

struct node \*temp;

temp=h;

if(temp->next==NULL)

{

free(temp);

h=NULL;

}

else

{

h=h->next;

printf("%d %s %s %s %f %d", temp->ssn, temp->name,temp->dept,

temp->desg,temp->sal, temp->phno );

free(temp);

}

count--;

return 0;

}

void main()

{

int ch,n,i;

h=NULL;

temp = temp1 = NULL;

printf("-----------------MENU--------------------\n");

printf("\n 1 - create a DLL of n emp");

printf("\n 2 - Display from beginning");

printf("\n 3 - Insert at end");

printf("\n 4 - delete at end");

printf("\n 5 - Insert at beg");

printf("\n 6 - delete at beg");

printf("\n 7 - exit\n");

printf("------------------------------------------\n");

while (1)

{

printf("\n Enter choice : ");

scanf("%d", &ch);

switch (ch)

{

case 1:

printf("\n Enter no of employees : ");

scanf("%d", &n);

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

insertend();

break;

case 2:

displaybeg();

break;

case 3:

insertend();

break;

case 4:

deleteend();

break;

case 5:

insertbeg();

break;

case 6:

deletebeg();

break;

case 7:

exit(0);

default:

printf("wrong choice\n");

}

}

}

11.Write a C program

1. To construct a binary search three of integers.
2. To traverse the tree using all the methods i.e, inorder, preorder and postorder.

#include<stdio.h>

#include<conio.h>

#include<alloc.h>

#include<process.h>

struct node1

{

int data;

struct node1 \*lt,\*rt;

};

typedef struct node1 node;

node \*root=NULL;

node \*insert(node \*,int);

void inorder(node \*);

void preorder(node \*);

void postorder(node \*);

int ch,num;

void main()

{

clrscr();

while(ch!=5)

{

printf("\n\*\*\*\*\*MENUOPERATION\*\*\*\*\*");

printf("\1\_INSERT\n");

printf("2\_INORDER\n");

printf("3\_POSTORDER\n");

printf("4\_PREORDER\n");

printf("5\_exit\n");

printf("enter ur choice:\n");

scanf("%d",&ch);

switch(ch)

{

case 1:printf("enter the elements to be inserted\n");

scanf("%d",&num);

root=insert(root,num);

break;

case 2:printf("elements in INORDER\n");

inorder(root);

break;

case 3:printf("elements in POSTORDER\n");

postorder(root);

break;

case 4:printf("elements in PREORDER\n");

preorder(root);

break;

case 5:exit(0);

default:printf("enter the ur choice\n");

}

}

getch();

}

node \*insert(node \*p,int num)

{

if(p==NULL)

{

p=(node\*)malloc(sizeof(node));

p->data=num;

p->rt=NULL;

p->lt=NULL;

}

else

{

if(num<p->data)

p->lt=insert(p->lt,num);

else

p->rt=insert(p->rt,num);

}

return(p);

}

void preorder(node \*p)

{

if(p!=NULL)

{

printf("%d\t",p->data);

preorder(p->lt);

preorder(p->rt);

}

}

void inorder(node \*p)

{

if(p!=NULL)

{

inorder(p->lt);

printf("%d\t",p->data);

inorder(p->rt);

}

}

void postorder(node \*p)

{

if(p!=NULL)

{

postorder(p->lt);

postorder(p->rt);

printf("%d\t",p->data);

}

}

12.I mplement hashing technique to map a given key K to the address space L. Resolve the collision

(if any) using linear probing

Given a File of N employee records with a set K of Keys(4-digit) which uniquely determine

the records in file F. Assume that file F is maintained in memory by a Hash Table(HT) of

m memory locations with L as the set of memory addresses (2-digit) of locations in HT.

Let the keys in K and addresses in L are Integers. Design and develop a program in C that

uses Hash function H: K → L as H(K)=K mod m (remainder method), and implement hashing technique to map a given key K to the address space L. Resolve the collision

(if any) using linear probing.

#include <stdio.h>

#include <stdlib.h>

#define MAX 100

/\*FUNCTION PROTOTYPE \*/

int create(int);

void linear\_prob(int[], int, int);

void display (int[]);

void main()

{

int a[MAX],num,key,i;

int ans=1;

printf(" collision handling by linear probing : \n");

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

{

a[i] = -1;

}

do

{

printf("\n Enter the data");

scanf("%4d", &num);

key=create(num);

linear\_prob(a,key,num);

printf("\n Do you wish to continue ? (1/0) ");

scanf("%d",&ans);

}while(ans);

display(a);

}

int create(int num)

{

int key;

key=num%100;

return key;

}

void linear\_prob(int a[MAX], int key, int num)

{

int flag, i, count=0;

flag=0;

if(a[key]== -1)

{

a[key] = num;

}

else

{

printf("\nCollision Detected...!!!\n");

i=0;

while(i<MAX)

{

if (a[i]!=-1)

count++;

i++;

}

printf("Collision avoided successfully using LINEAR PROBING\n");

if(count == MAX)

{

printf("\n Hash table is full");

display(a);

exit(1);

}

for(i=key+1; i<MAX; i++)

if(a[i] == -1)

{

a[i] = num;

flag =1;

break;

}

//for(i=0;i<key;i++)

i=0;

while((i<key) && (flag==0))

{

if(a[i] == -1)

{

a[i] = num;

flag=1;

break;

}

i++;

}

}

}

void display(int a[MAX])

{

int i,choice;

printf("1.Display ALL\n 2.Filtered Display\n");

scanf("%d",&choice);

if(choice==1)

{

printf("\n the hash table is\n");

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

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

}

else

{

printf("\n the hash table is\n");

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

if(a[i]!=-1)

{

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

continue;

}

}

}