***WEEK 2 DS PRELAB PROGRAMS***

***PRE-LAB QUESTIONS:***

1).Define Data structure .Differentiate between a linear and non linear Data structures.

A.Data structure is a data organization ,management and storage format that enables efficient access and modification.It is a collection of data values ,the relationships among them ,and the functions or operations that can be applied to data.

In a linear data structure, data elements are arranged in a linear order where each and every elements are attached to its previous and next adjacent.eg: arrays,linked-lists,stacks,queues.

In a non-linear data structure, data elements are attached in hierarchically manner.eg: trees,graphs.

2)Give examples of linear and non linear datatypes?

A)Linear: Arrays, linked lists, stacks, queues.

Non-Linear: Trees,graphs

3)What is the need for arrays? Give the advantages of arrays.

A. Arrays are used when there is need to use many variables of the same type. It can be defined as a sequence of objects which are of the same data type.

ADVANTAGES OF ARRAYS:

1.Arrays represent multiple data items of the same type using a single name.

2.In arrays, the elements can be accessed randomly by using the index number.

3.Arrays allocate memory in contiguous memory locations for all its elements. Hence there is no chance of extra memory being allocated in case of arrays. This avoids memory overflow or shortage of memory in arrays.

4)What are the disadvantages of an array?

1. The number of elements to be stored in an array should be known in advance.

2.An array is a static structure (which means the array is of fixed size). Once declared the size of the array cannot be modified. The memory which is allocated to it cannot be increased or decreased.

3.Insertion and deletion are quite difficult in an array as the elements are stored in consecutive memory locations and the shifting operation is costly.

4.Allocating more memory than the requirement leads to wastage of memory space and less allocation of memory also leads to a problem.

5) C function to merge contents of 2 arrays.

A. void merge(int a[], int m, int b[], int n, int c[]) {

int i, j, k;

j = k = 0;

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

if (j < m && k < n) {

if (a[j] < b[k]) {

c[i] = a[j];

j++;

}

else {

c[i] = b[k];

k++;

}

i++;

}

else if (j == m) {

for (; i < m + n;) {

c[i] = b[k];

k++;

i++;

}

}

else {

for (; i < m + n;) {

c[i] = a[j];

j++;

i++;

}

}

}

}

1. write the alogorithm for the function to swap the elements of an array as mentioned

F1:3 4 5 6 2 8 1

F2:4 3 6 5 8 2 1

A.

step #1: Read array

step #2: count no of elements are present in the array

step #3: set a counter variable from 0 to count/2 with increment of 2.

step #4: Now, swap array[counter] and array[counter+1]

step #5: print the array

***Prelab Programs:***

***1.Formula based representation.***

#include <stdio.h>

*int* a[20],n,i,j,pos,item,temp,toBeSearched,found;

//Inserting

*void* Insert(*int* *a*[],*int* *n*)

{

    printf("Type a Position where you want to insert: ");

    scanf("%d",&pos);

    printf("Enter Element to be inserted: ");

    scanf("%d",&item);

    printf("Original Elements are:\n");

    for(i=0;i<*n*;i++)

    {

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

    }

    j = *n*;

*n* = *n*+1;

    while(j >= pos-1)

    {

*a*[j+1] = *a*[j];

        j--;

    }

*a*[pos-1] = item;

    printf("Elements after Insertion are:\n");

    for(i=0;i<*n*;i++)

    {

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

    }

    printf("Insertion Successful\n\n");

}

//Deleting

*void* Delete(*int* *a*[],*int* *n*)

{

    if(*n*==0)

        printf("ERROR: Can't Delete Because their are no elements in the array");

    printf("Type the Position to be deleted: ");

    scanf("%d",&pos);

    printf("Original Elements are:\n");

    for(i=0;i<*n*;i++)

    {

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

    }

    j = pos;

    while(j < *n*)

    {

*a*[j-1] = *a*[j];

        j++;

    }

*n* = *n* - 1;

    printf("Elements After Deletion:\n");

    for(i=0;i<*n*;i++)

    {

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

    }

    printf("Deletion Successful\n\n");

}

//Sorting

*void* Sort(*int* *a*[],*int* *n*)

{

    if(*n*==0)

        printf("ERROR: Can't Sort Because their are no elements in the array");

    else

    {

        printf("Original Elements are:\n");

        for(i=0;i<*n*;i++)

        {

            printf("a[%d] = %d\n",i,*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("Elements After Sorting\n");

        for(i=0;i<*n*;i++)

        {

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

        }

        printf("Sorted Successfully\n\n");

    }

}

//Searching

*void* Search(*int* *a*[],*int* *n*)

{

    if(*n*==0)

        printf("ERROR: Can't Search Because their are no elements in the array");

    else

    {

        printf("Enter element to be searched: ");

        scanf("%d",&toBeSearched);

        found = 0;

        for(i=0;i<*n*;i++)

        {

            if(toBeSearched == *a*[i])

            {

                found = 1;

                printf("%d found at %d\n\n",toBeSearched,i+1);

            }

        }

        if(found == 0)

            printf("%d not found\n\n",toBeSearched);

    }

}

//Update

*void* Update(*int* *a*[],*int* *n*)

{

    if(*n*==0)

        printf("ERROR: Can't Update Because their are no elements in the array");

    else

    {

        printf("Enter postion to be updated: ");

        scanf("%d",&pos);

        printf("Enter Element to be placed: ");

        scanf("%d",&item);

*a*[pos-1] = item;

        printf("Original Elements are:\n");

        for(i=0;i<*n*;i++)

        {

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

        }

        printf("Updated Successfully\n");

    }

}

*void* main()

{

*int* opt;

*char* choice;

    printf("Enter size: ");

    scanf("%d",&n);

    printf("Enter elements of array: ");

    for(i=0;i<n;i++)

    {

        scanf("%d",&a[i]);

    }

    do

    {

        printf(" \t\t1) Insertion \t\t\n \t\t2) Deletion \t\t\n \t\t3) Sorting \t\t\n \t\t4) Searching \t\t\n \t\t5) Updating \t\t\n\nEnter an option: ");

        scanf("%d",&opt);

        switch(opt)

        {

            case 1: Insert(a,n);

                    break;

            case 2: Delete(a,n);

                    break;

            case 3: Sort(a,n);

                    break;

            case 4: Search(a,n);

                    break;

            case 5: Update(a,n);

                    break;

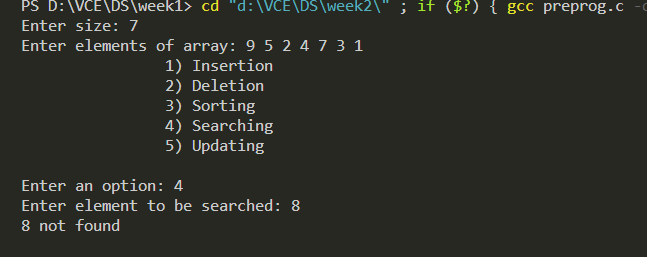
        }

        printf("Do You Want To do the Operations Again[Y/N]: ");

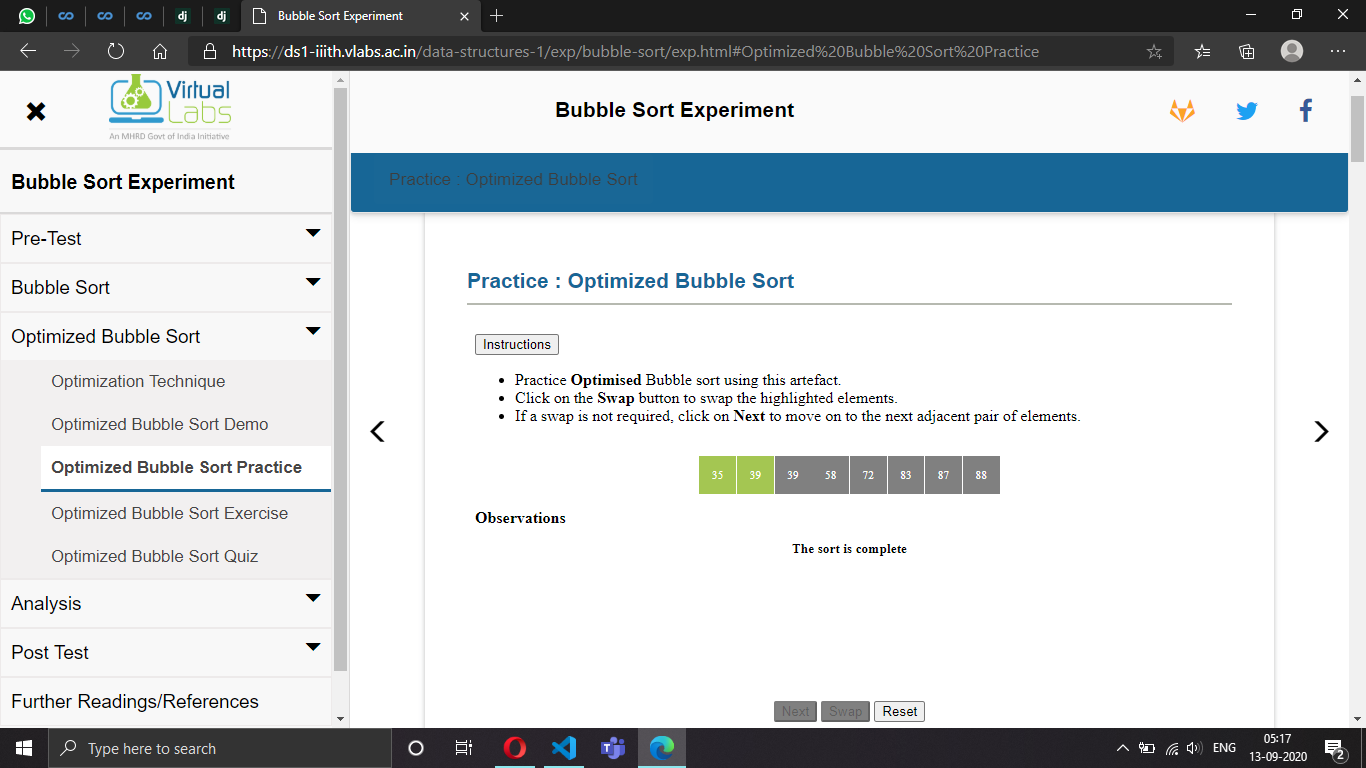
        scanf("%c",&choice);

    }while(choice=='Y' || choice=='y');

}



Bubble sort - Practice and simulation



**Lab Programs:**

1. A User is interested in buying some items from online shopping website.

He tried to add items in to the cart. He has with him a valid amount of only

5000 /-. If items exceeded in the cart, he has to delete the items. Implement a

program to determine the following.

Consider attributes of the cart as item id, item name, Quantity and cost.

 Adding items into the cart.

 Delete items from the cart.

 Display items in the cart.

 Search if a specific item is present or not.

 Arrange items in the descending order of the cost.

#include <stdio.h>

#include<stdlib.h>

#include<string.h>

*int* limit;

*struct* Shopping

{

*int* id,quantity,cost;

*char* name[32];

};

*int* size(*struct* Shopping \**cart*){

*int* i=0,count=0;

    while(*cart*[i].id!=-1)

    {

        count++;

        i++;

    }

    return count;

}

*void* DisplayItems(*struct* Shopping \**cart*){

*int* i;

    printf("Elements in the Cart are:\n");

    for(i=0;i<size(*cart*);i++){

        printf("%d",i);

        printf("Id: %d    Name: %s    quantity: %d    Cost: %d\n",*cart*[i].id,*cart*[i].name,*cart*[i].quantity,*cart*[i].cost);

    }

}

*void* DeleteItem(*int* *pos*,*struct* Shopping\* *cart*){

*int* i,j;

    for(j=0;j<size(*cart*);j++)

    {

        if(*cart*[j].id==*pos*)

        {

*pos*=j+1;

            for(i=*pos*-1;i<size(*cart*);i++)

            {

*cart*[i].id=*cart*[i+1].id;

                strcpy(*cart*[i].name,*cart*[i+1].name);

*cart*[i].quantity=*cart*[i+1].quantity;

*cart*[i].cost=*cart*[i+1].cost;

            }

         }

     }

     printf("Deleted Successfully\n");

     if(size(*cart*)==0)

         printf("Cart is Empty");

     else

         DisplayItems(*cart*);

}

*int* cost(*struct* Shopping\* *cart*){

*int* i,t=0;

    for(i=0;i<size(*cart*);i++){

        t=t+*cart*[i].cost;

    }

    return t;

}

*void* AddItems(*struct* Shopping *temp*,*struct* Shopping \**cart*){

*int* l,p;

*char* command;

    if((cost(*cart*)+*temp*.cost)>limit)

    {

        printf("ERROR: Budget Exceeding %d Rupees. Please remove some item(s).\n",limit);

        printf("Do you want to remove now?(Y/N)");

        scanf(" %c",&command);

        if(command=='Y' || command=='y')

        {

            printf("Enter ID of the element to be Deleted: ");

            scanf("%d",&p);

            DeleteItem(p,*cart*);

        }

    }

    else

    {

        l=size(*cart*);

*cart*[l].id=*temp*.id;

        strcpy(*cart*[l].name,*temp*.name);

*cart*[l].quantity=*temp*.quantity;

*cart*[l].cost=*temp*.cost;

        DisplayItems(*cart*);

    }

}

*void* SearchByID(*int* *x*,*struct* Shopping\* *cart*){

*int* i=0;

    if(size(*cart*)==0)

    {

        printf("No items in cart to search");

    }

    else

    {

        while(i<size(*cart*))

        {

            if(*cart*[i].id==*x*)

            {

                printf("Id: %d\tName: %s\tquantity: %d\t\tCost: %d\n",*cart*[i].id,*cart*[i].name,*cart*[i].quantity,*cart*[i].cost);

            }

            i++;

        }

    }

}

*void* SortByCost(*struct* Shopping\* *cart*)

{

*int* i,j;

*struct* Shopping temp;

    if(size(*cart*)==0)

    {

        printf("No Items in cart to sort");

    }

    else

    {

        for(i=0;i<size(*cart*);i++)

        {

            for(j=i+1;j<size(*cart*);j++)

            {

                if(*cart*[j].cost<*cart*[i].cost)

                {

                    temp.id=*cart*[j].id;

*cart*[j].id=*cart*[i].id;

*cart*[i].id=temp.id;

                    strcpy(temp.name,*cart*[i].name);

                    strcpy(*cart*[i].name,*cart*[j].name);

                    strcpy(*cart*[j].name,temp.name);

                    temp.quantity=*cart*[j].quantity;

*cart*[j].quantity=*cart*[i].quantity;

*cart*[i].quantity=temp.quantity;

                    temp.cost=*cart*[j].cost;

*cart*[j].cost=*cart*[i].cost;

*cart*[i].cost=temp.cost;

                }

            }

        }

        printf("Sorted Successfully\n");

    }

}

*void* main()

{

    printf("Enter your budget: ");

    scanf("%d",&limit);

*struct* Shopping temp,cart[20];

*int* i,item,element,position,opt;

*char* command='y';

    for(i=0;i<20;i++)

        cart[i].id=-1;

    while(command=='y')

    {

        printf("Enter an option from below:\n ");

        printf("\t1) Add Items\n");

        printf("\t2) Delete Items\n");

        printf("\t3) Display them in Descending order based on Cost\n");

        printf("\t4) Search Item\n");

        printf("\t5) Display all The Items\n");

        scanf("%d",&opt);

        switch(opt){

            case 1: printf("Enter the Details of the Item : \n\n");

                    printf("Enter name of Item: ");

                    scanf("%s",temp.name);

                    printf("Enter ID of Item: ");

                    scanf("%d",&temp.id);

                    printf("Enter Quantity: ");

                    scanf("%d",&temp.quantity);

                    printf("Enter cost of Item: ");

                    scanf("%d",&temp.cost);

                    AddItems(temp,cart);

                    break;

            case 2: if(size(cart)==0)

                    {

                        printf("ERROR: No Items in Cart\n");

                    }

                    else

                    {

                        printf("Enter ID of element to be Deleted: ");

                        scanf("%d",&position);

                        DeleteItem(position,cart);

                    }

                    break;

            case 3: printf("Enter Item ID to be Searched: ");

                    scanf("%d",&element);

                    SearchByID(element,cart);

                    break;

            case 4: SortByCost(cart);

                    DisplayItems(cart);

                    break;

            case 5: DisplayItems(cart);

                    break;

        }

        printf("\nDo you want to Still Add Items?(Y/N):");

        scanf(" %c",&command);

        if(command=='N' || command=='n'){

            printf("\tEnd");

        }

    }

}

2.Spiral Matrix form.

#include<stdlib.h>

#include<stdio.h>

*void* main()

{

*int* i,j,n,\*\*a;

    printf("Enter size of the square matrix : ");

    scanf("%d",&n);

    a=(*int* \*\*)calloc(n,sizeof(*int* \*));

    printf("Enter elements : ");

    for(i=0;i<n;i++)

    {

        a[i]=(*int* \*)calloc(n,sizeof(*int*));

        for(j=0;j<n;j++)

            scanf("%d",&a[i][j]);

    }

*int* m=n,k=0,l=0;

    while(k<m&&l<n)

    {

        for(i=l;i<n;++i)

            printf("%d ",a[k][i]);

        k++;

        for(j=k;j<m;++j)

            printf("%d ",a[j][n-1]);

        n--;

        if(k<m)

        {

            for(i=n-1;i>=l;--i)

                printf("%d ",a[m-1][i]);

            m--;

        }

        if(l<n)

        {

            for(j=m-1;j>=k;--j)

                printf("%d ",a[j][l]);

            l++;

        }

    }

}

