# A picture containing text, clipart, vector graphics Description automatically generatedl DSA LAB REPORT

# Section-F11-SLOT2

|  |  |  |  |
| --- | --- | --- | --- |
| **EXERCISE NO:** | **9** | **DATE OF EXERCISE:** | **16.11.2021** |
| **ROLL NUMBER:** | **20051699** | **GROUP NO.:** | **21111** |
| **NAME IN CAPITAL :** | **KRISHNA BISWAL** | | |

Lab Assignment (LA):

**Q1.** **WAP to sort an array of n integers in an ascending order using Heap sort.**

HANDWRITTEN CODE:

SOURCECODE

#include <stdio.h>

void main()

{

int heap[10], no, i, j, c, root, temp;

printf("\nEnter no of elements :");

scanf("%d", &no);

printf("\nEnter the nos : ");

for (i = 0; i < no; i++)

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

for (i = 1; i < no; i++)

{

c = i;

do

{

root = (c - 1) / 2;

if (heap[root] < heap[c])

{

temp = heap[root];

heap[root] = heap[c];

heap[c] = temp;

}

c = root;

} while (c != 0);

}

printf("\nHeap array : ");

for (i = 0; i < no; i++)

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

for (j = no - 1; j >= 0; j--)

{

temp = heap[0];

heap[0] = heap[j];

heap[j] = temp;

root = 0;

do

{

c = 2 \* root + 1;

if ((heap[c] < heap[c + 1]) && c < j-1)

c++;

if (heap[root]<heap[c] && c<j)

{

temp = heap[root];

heap[root] = heap[c];

heap[c] = temp;

}

root = c;

} while (c < j);

}

printf("\nThe sorted array is : ");

for (i = 0; i < no; i++)

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

}

OUTPUT

**Q2.** **WAP to sort an array of n integers in an ascending order using merge sort.**

**HANDWRITTEN CODE:**

SOURCECODE

#include<stdio.h>

void merge(int arr[],int min,int mid,int max)

{

int tmp[30];

int i,j,k,m;

j=min;

m=mid+1;

for(i=min; j<=mid && m<=max ; i++)

{

if(arr[j]<=arr[m])

{

tmp[i]=arr[j];

j++;

}

else

{

tmp[i]=arr[m];

m++;

}

}

if(j>mid)

{

for(k=m; k<=max; k++)

{

tmp[i]=arr[k];

i++;

}

}

else

{

for(k=j; k<=mid; k++)

{

tmp[i]=arr[k];

i++;

}

}

for(k=min; k<=max; k++)

arr[k]=tmp[k];

}

void sortm(int arr[],int min,int max)

{

int mid;

if(min<max)

{

mid=(min+max)/2;

sortm(arr,min,mid);

sortm(arr,mid+1,max);

merge(arr,min,mid,max);

}

}

int main()

{

int arr[30];

int i,size;

printf("\tMerge sort\n");

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

printf("How many numbers you want to sort?: ");

scanf("%d",&size);

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

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

{

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

}

sortm(arr,0,size-1);

printf("\nSorted elements after using merge sort:\n\n");

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

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

return 0;

}

OUTPUT

**Q3.** **WAP to sort an array of n doubles in a descending order using quick sort.**

**HANDWRITTEN CODE :**

**SOURCE CODE**

#include<stdio.h>

void quicksort(double number[25],int first,int last){

int i, j, pivot, temp;

if(first<last){

pivot=first;

i=first;

j=last;

while(i<j){

while(number[i]>=number[pivot]&&i<last)

i++;

while(number[j]<number[pivot])

j--;

if(i<j){

temp=number[i];

number[i]=number[j];

number[j]=temp;

}

}

temp=number[pivot];

number[pivot]=number[j];

number[j]=temp;

quicksort(number,first,j-1);

quicksort(number,j+1,last);

}

}

int main(){

int i,count;

double number[25];

printf("Enter some elements (Max. - 25): ");

scanf("%d",&count);

printf("Enter %d elements: ", count);

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

scanf("%lf",&number[i]);

quicksort(number,0,count-1);

printf("The Sorted Order is: ");

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

printf(" %lf",number[i]);

return 0;

}

OUTPUT

**Q4.** **WAP to sort an array of n integers in a descending order using insertion sort.**

**HANDWRITTEN CODE :**

**SOURCE CODE**

#include <stdio.h>

int main()

{

int n, i, j, temp;

int arr[64];

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

scanf("%d", &n);

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

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

{

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

}

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

{

j = i;

while ( j > 0 && arr[j-1] < arr[j])

{

temp = arr[j];

arr[j] = arr[j-1];

arr[j-1] = temp;

j--;

}

}

printf("Sorted list in descending order:\n");

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

{

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

}

return 0;

}

OUTPUT

**Q5.** **WAP to store n floats in linked list and sort them using selection sort.**

**HANDWRITTEN CODE :**

**SOURCE CODE**

#include<stdio.h>

#include<stdlib.h>

typedef struct Node

{

float data;

struct Node \*link;

}node;

node \*head = NULL;

void print();

void swap(node \*p1, node\*p2);

void SelectionSort(node \*head);

void insert(float data, float position);

float main()

{

insert(4,1);

insert(2,2);

insert(3,3);

insert(1,4);

insert(0,5);

printf("\n Before sorting = ");

print();

SelectionSort(head);

printf("\n After sorting = ");

print();

return 0;

}

void SelectionSort(node \*head)

{

node \*start = head;

node \*traverse;

node \*min;

while(start->link)

{

min = start;

traverse = start->link;

while(traverse)

{

if( min->data > traverse->data )

{

min = traverse;

}

traverse = traverse->link;

}

swap(start,min);

start = start->link;

}

}

void swap(node \*p1, node\*p2)

{

float temp = p1->data;

p1->data = p2->data;

p2->data = temp;

}

void insert(float data, float position)

{

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

temp->data = data;

temp->link = NULL;

if(position==1)

{

temp->link = head;

head = temp;

return ;

}

node \*traverse = head;

float i;

for(i=0; i<position-2; i++)

{

traverse = traverse->link;

}

temp->link = traverse->link;

traverse->link = temp;

}

void print()

{

node \*p = head;

while(p)

{

printf(" %f",p->data);

p = p->link;

}

printf(" \n\n");

}

OUTPUT

**Q6.** **WAP to store n floats in linked list and sort them using bubble sort.**

**HANDWRITTEN CODE :**

**SOURCE CODE**

#include<stdio.h>

#include<stdlib.h>

struct Node

{

float data;

struct Node \*next;

};

void insertAtTheBegin(struct Node \*\*start\_ref, float data);

void bubbleSort(struct Node \*start);

void swap(struct Node \*a, struct Node \*b);

void printList(struct Node \*start);

int main()

{

float arr[] = {1.1, 1.2, 2.3, 1.4, 1.5, 9};

float list\_size;

int i;

struct Node \*start = NULL;

for (i = 0; i< 6; i++)

insertAtTheBegin(&start, arr[i]);

printf("\nLinked list before sorting : \n");

printList(start);

bubbleSort(start);

printf("\n\nLinked list after sorting : \n");

printList(start);

getchar();

return 0;

}

void insertAtTheBegin(struct Node \*\*start\_ref, float data)

{

struct Node\* ptr1 = (struct Node\*)malloc(sizeof(struct Node));

ptr1->data = data;

ptr1->next = \*start\_ref;

\*start\_ref = ptr1;

}

void printList(struct Node \*start)

{

struct Node \*temp = start;

printf("\n");

while (temp!=NULL)

{

printf("%f ", temp->data);

temp = temp->next;

}

}

void bubbleSort(struct Node \*start)

{

float swapped, i;

struct Node \*ptr1;

struct Node \*lptr = NULL;

if (start == NULL)

return;

do

{

swapped = 0;

ptr1 = start;

while (ptr1->next != lptr)

{

if (ptr1->data > ptr1->next->data)

{

swap(ptr1, ptr1->next);

swapped = 1;

}

ptr1 = ptr1->next;

}

lptr = ptr1;

}

while (swapped);

}

void swap(struct Node \*a, struct Node \*b)

{

float temp = a->data;

a->data = b->data;

b->data = temp;

}

OUTPUT

|  |  |
| --- | --- |
| DECLARATION  I hereby declare that,  ☑ I have written the assignment in my own handwritting as mentioned in Handwritten  Code Section.  ☑ I have typed my source code in code editor and taken my own test case output after  running of code .  Full Signature of the Student   |  | | --- | | Signature | |