**PROGRAM-4**

4. Write an algorithm and program to sort n numbers using Selection sort technique

i) Using arrays

#include <stdio.h>

void selection\_sort();

int a[30], n;

void main()

{

int i;

printf("\nEnter size of an array: ");

scanf("%d", &n);

printf("\nEnter elements of an array:\n");

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

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

selection\_sort();

printf("\n\nAfter sorting:\n");

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

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

getch();

}

void selection\_sort()

{

int i, j, min, temp;

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

{

min = i;

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

{

if (a[j] < a[min])

min = j;

}

temp = a[i];

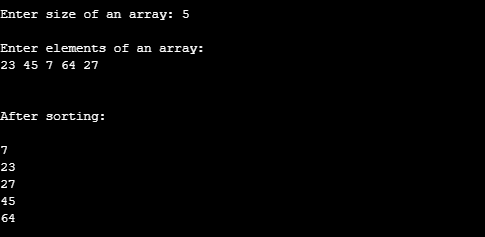
a[i] = a[min];

a[min] = temp;

}

}

**OUTPUT-**



ii) Using recursion

#include <stdio.h>

void selection(int [], int, int, int, int);

int main()

{

int list[30], size, temp, i, j;

printf("Enter the size of the list: ");

scanf("%d", &size);

printf("Enter the elements in list:\n");

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

{

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

}

selection(list, 0, 0, size, 1);

printf("The sorted list in ascending order is\n");

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

{

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

}

return 0;

}

void selection(int list[], int i, int j, int size, int flag)

{

int temp;

if (i < size - 1)

{

if (flag)

{

j = i + 1;

}

if (j < size)

{

if (list[i] > list[j])

{

temp = list[i];

list[i] = list[j];

list[j] = temp;

}

selection(list, i, j + 1, size, 0);

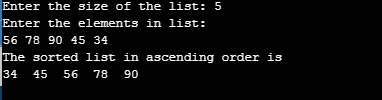
}

selection(list, i + 1, 0, size, 1);

}

}

**OUTPUT-**



iii) using recursion and linked list-

#include <bits/stdc++.h>

using namespace std;

struct Node {

int data;

struct Node\* next;

};

void swapNodes(struct Node\*\* head\_ref, struct Node\* currX,

struct Node\* currY, struct Node\* prevY)

{

\*head\_ref = currY;

prevY->next = currX;

struct Node\* temp = currY->next;

currY->next = currX->next;

currX->next = temp;

}

struct Node\* recurSelectionSort(struct Node\* head)

{

if (head->next == NULL)

return head;

struct Node\* min = head;

struct Node\* beforeMin = NULL;

struct Node\* ptr;

for (ptr = head; ptr->next != NULL; ptr = ptr->next) {

if (ptr->next->data < min->data) {

min = ptr->next;

beforeMin = ptr;

}

}

if (min != head)

swapNodes(&head, head, min, beforeMin);

head->next = recurSelectionSort(head->next);

return head;

}

void sort(struct Node\*\* head\_ref)

{

if ((\*head\_ref) == NULL)

return;

\*head\_ref = recurSelectionSort(\*head\_ref);

}

void push(struct Node\*\* head\_ref, int new\_data)

{

struct Node\* new\_node =

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

new\_node->data = new\_data;

new\_node->next = (\*head\_ref);

(\*head\_ref) = new\_node;

}

void printList(struct Node\* head)

{

while (head != NULL) {

cout << head->data << " ";

head = head->next;

}

}

int main()

{

struct Node\* head = NULL;

push(&head, 6);

push(&head, 4);

push(&head, 8);

push(&head, 12);

push(&head, 10);

cout << "Linked list before sorting:n";

printList(head);

sort(&head);

cout << "\nLinked list after sorting:n";

printList(head);

return 0;

}

