BABASAHEB BHIMRAO AMBEDKAR UNIVERSITY,LUCKNOW

2024-25

COMPUTER ENGINEERING

Data Structure lab

ECS-301

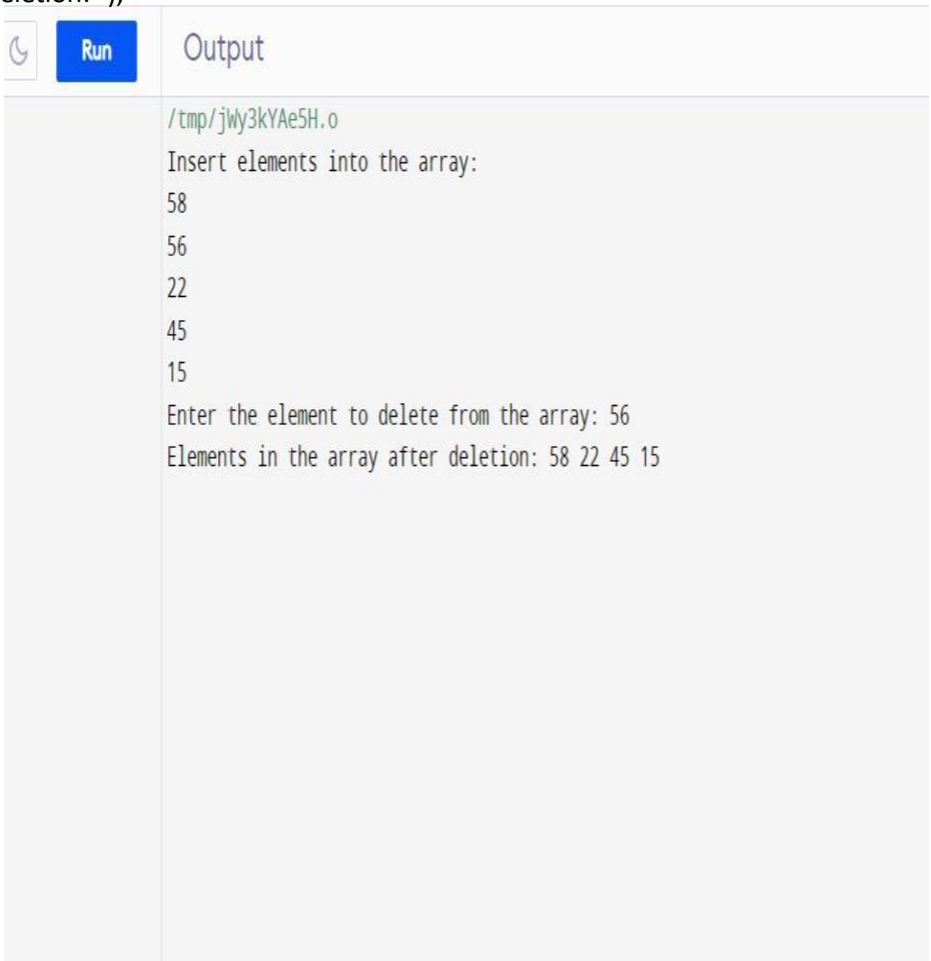
SUBMITTED BY:

Mr. Ram Singar Verma

NAME : AVNISH SRIVASTAVA
ROLL NO. : 238209
BRANCH : COMPUTER ENGINEERING
YEAR : 2ND
SUBJECT : DATA STRUCTURES USING C

Write a program to insertion and deletion of element on array.

```
#include <stdio.h>
int main() {
   int i, a[5], n;

   printf("Insert elements into the array:\n");
   for (i = 0; i < 5; i++) {
      scanf("%d", &a[i]);
   }

   printf("Enter the element to delete from the array: ");
   scanf("%d", &n);

   printf("Elements in the array after deletion: ");
   for (i = 0; i < 5; i++) {
      if (n != a[i]) {

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

   printf("\n");
   return 0;
}
```



Write a program to implement Stack operation using an array.

```
#include <stdio.h>

#define MAX_SIZE 10

int stack[MAX_SIZE];

int top = -1;


void push(int element) {
   if (top == MAX_SIZE - 1) {
      printf("Stack Overflow\n");
   } else {
      stack[++top] = element;
      printf("Element %d pushed to stack\n", element);
   }
}


int pop() {
   if (top == -1) {
      printf("Stack Underflow\n");
      return -1;
   } else {
      int element = stack[top--];
      printf("Element %d popped from stack\n", element);
      return element;
   }
}


void display() {
   if (top == -1) {
      printf("Stack is empty\n");
```

```
    } else {
        printf("Elements in the stack:\n");
        for (int i = top; i >= 0; i--) {
            printf("%d\n", stack[i]);
        }
    }
}

int main() {
    int choice, element;

    do {
        printf("\nStack Operations:\n");
        printf("1. Push\n");
        printf("2. Pop\n");
        printf("3. Display\n");
        printf("4. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                printf("Enter the element to push: ");
                scanf("%d", &element);
                push(element);
                break;
            case 2:
                element = pop();
                if (element != -1) {
```

```
            printf("Popped element: %d\n", element);
        }
        break;
    case 3:
        display();
        break;
    case 4:
        printf("Exiting...\n");
        break;
    default:
        printf("Invalid choice\n");
        break;
  }
 } while (choice != 4);

 return 0;
}
```

```
/tmp/UHpNKGNBvb.o
Stack Operations:
1. Push
2. Pop
3. Display
4. Exit
Enter your choice: 1
Enter the element to push: 6
Element 6 pushed to stack

Stack Operations:
1. Push
2. Pop
3. Display
4. Exit
Enter your choice: 2
Element 6 popped from stack
Popped element: 6

Stack Operations:
1. Push
2. Pop
3. Display
4. Exit
Enter your choice: 3
```

Write a program to evaluations of postfix arithmetic expression.

```
#include <stdio.h>

int main() {
   int array[100], n, c, d, swap;

   printf("Enter the number of elements: ");
   scanf("%d", &n);

   printf("Enter %d integers:\n", n);
   for (c = 0; c < n; c++) {
      scanf("%d", &array[c]);
   }

   // Bubble sort
   for (c = 0; c < (n - 1); c++) {
      for (d = 0; d < (n - c - 1); d++) {
         if (array[d] > array[d + 1]) {
            // Swap elements
            swap = array[d];
            array[d] = array[d + 1];
            array[d + 1] = swap;
         }
      }
   }

   printf("Sorted list in ascending order:\n");
   for (c = 0; c < n; c++) {
      printf("%d\n", array[c]);
```

```
    }

    return 0;
}
```

Output

```
/tmp/Sc4paZ20bZ.o
Enter the number of elements: 6
Enter 6 integers:
23
44
72
09
33
108
Sorted list in ascending order:
9
23
33
44
72
108
```

Write a program to implement Heap Sort algorithm.

```
#include <stdio.h>

void heapsort(int[], int);
void heapify(int[], int);
void adjust(int[], int);

int main() {
   int n, i, a[50];

   printf("Enter the limit: ");
   scanf("%d", &n);

   printf("Enter the elements: ");
   for (i = 0; i < n; i++)
      scanf("%d", &a[i]);

   heapsort(a, n);

   printf("\nThe Sorted Elements Are:\n");
   for (i = 0; i < n; i++)
      printf("\t%d", a[i]);
   printf("\n");

   return 0;
}

void heapsort(int a[], int n) {
   int i, t;
```

```
    heapify(a, n);

    for (i = n - 1; i > 0; i--) {
        t = a[0];
        a[0] = a[i];
        a[i] = t;
        adjust(a, i);
    }
}

void heapify(int a[], int n) {
    int k, i, j, item;

    for (k = 1; k < n; k++) {
        item = a[k];
        i = k;
        j = (i - 1) / 2;

        while ((i > 0) && (item > a[j])) {
            a[i] = a[j];
            i = j;
            j = (i - 1) / 2;
        }

        a[i] = item;
    }
}
```

```
void adjust(int a[], int n) {
    int i, j, item;

    j = 0;
    item = a[j];
    i = 2 * j + 1;

    while (i <= n - 1) {
        if (i + 1 <= n - 1 && a[i] < a[i + 1])
            i++;

        if (item < a[i]) {
            a[j] = a[i];
            j = i;
            i = 2 * j + 1;
        } else {
            break;
        }
    }

    a[j] = item;
}
```

Output

```
/tmp/Sc4paZ20bZ.o
Enter the limit: 8
Enter the elements: 4
6
7
9
6
5
4

5
The Sorted Elements Are:
    4   4   5   5   6   6   7   9
```

Write a program to implement Insertion Sort algorithm

```
#include<stdio.h>
int main(){
 int i,j,s,temp,a[20];
 printf("Enter total elements: ");
 scanf("%d",&s);
 printf("Enter %d elements: ",s);
 for(i=0;i<s;i++)
 scanf("%d",&a[i]);
 for(i=1;i<s;i++){
 temp=a[i];
 j=i-1;
 while((temp<a[j])&&(j>=0)){
 a[j+1]=a[j];
 j=j-1;
 }
 a[j+1]=temp;
 }
 printf("After sorting: ");
 for(i=0;i<s;i++)
 printf(" %d",a[i]);
 return 0;
}
```

Output

```
/tmp/Sc4paZ20bZ.o
Enter total elements: 4
Enter 4 elements: 3
8
0
56
After sorting:  0 3 8 56
```