

11/10/24

## Data Structure Observation

- 1) Write a program to simulate the working of stack using an array with the following:
- a) Push                      (b) Pop                      (c) Display

The program should print appropriate messages for stack overflow, stack underflow.

Pseudocode

Set length of array as max and top index as top - 1.

- 1) Create an empty stack
- 2) Push() - Push an item into the stack.  $top = top + 1$   
 $s[top] = item$   
 If  $top = max - 1$ , stack overflow
- 3) Pop() - Remove the item that was inserted most recently.  $\Rightarrow item = s[top] \Rightarrow top = top - 1$
- 4) Display() - Display stack items.  
 If  $(top \geq -1)$   
 { printf("Stack is empty");  
 return; }  
 for  $(i = top; i \geq 0; i--)$   
 printf("%d\n", s[i]);  
 }

- 1) Set length of array as max and top index as top - 1
- 2) Create an empty stack
- 3) void push()
  - if  $(top = max - 1)$
  - printf("Stack overflow");
  - return;
  - $top++$ ;
  - scanf("%d", item)
  - $s[top] = item$

```

4) int pop() {
    if (top == -1) {
        pf("Underflow")
        return -1
    }
    item = s[top]
    top = top - 1
    return (item)
}

```

```

5) void display() {
    if (top == -1) {
        ("Stack empty")
    }
    else
        for (i = top; i > -1; i--)
            pf("%d\n", stack[i]);
}

```

Code:

```

#include <stdio.h>
int s[10], top = -1, i, item, ch;
void main() {
    while(1) {
        printf("Enter\n 1. Push\n 2. Pop\n 3. Display\n 4. Exit");
        scanf("%d", &ch);
        switch(ch) {
            case 1: push();
                    break;
            case 2: item = pop();
                    if (item != -1) {
                        printf("Popped element is: %d\n", item);
                    }
                    break;
        }
    }
}

```

```

    case 3: display();
            break;
    case 4: exit(0);
            break;
}
}

```

```

void push() {
    if (top == max - 1) {
        printf("Stack overflow\n");
    }
    else {
        top++;
        printf("Enter element to push");
        scanf("%d", &item);
        s[top] = item;
    }
}

```

```

int pop() {
    if (top == -1) {
        printf("Stack underflow\n");
        return -1;
    }
    item = s[top];
    top = top - 1;
    return (item);
}

```

```

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

```



else {

printf("Stack is\n");

for (i = top; i >= 0; i--) {

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

}

}

Output:

Enter

- 1- Push
- 2- Pop
- 3- Display
- 4- Exit

1

Enter element to push: 11

Enter

- 1- Push
- 2- Pop
- 3- Display
- 4- Exit

1

Enter element to push: 22

Enter

- 1- Push
- 2- Pop
- 3- Display
- 4- Exit

1

Enter element to push: 33

Enter

1. Push
  2. Pop
  3. Display
  4. Exit
- 3

The stack is :

33

22

11

Enter

1. Push
  2. Pop
  3. Display
  4. Exit
- 1

Stack overflow

Enter

1. Push
  2. Pop
  3. Display
  4. Exit
- 2

Popped element is : 33

Enter

1. Push
  2. Pop
  3. Display
  4. Exit
- 2

Popped element is : 22

Enter

1. Push
2. Pop
3. Display
4. Exit

2

Popped element is = 11

Enter

1. Push
2. Pop
3. Display
4. Exit

2

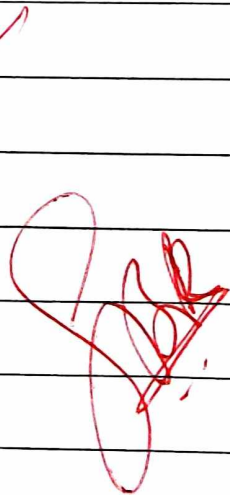
Stack underflow

\$

Enter

1. Push
2. Pop
3. Display
4. Exit

4

 01/10/2024