Aim:

To implement stack operations (push, pop, display) and check whether the stack is valid.

Algorithm:

1. Initialize an empty stack with max size.
2. Push: If stack is not full, add element at top.
3. Pop: If stack is not empty, remove element from top.
4. Display: Print elements from top to bottom.
5. Validate: Check overflow/underflow conditions.

C program:

```
#include <stdio.h>
#define MAX 5

int stack[MAX], top=-1;

void push(int x) {
   if(top==MAX-1) printf("Stack Overflow!\n");
   else stack[++top]=x;
}

void pop() {
   if(top==-1) printf("Stack Underflow!\n");
   else printf("Popped: %d\n", stack[top--]);
}

void display() {
   if(top==-1) printf("Stack Empty\n");
   else {
       printf("Stack elements: ");
       for(int i=top;i>=0;i--) printf("%d ",stack[i]);
       printf("\n");
   }
}

int main() {
   push(10); push(20); push(30);
   display();
   pop();
   display();
   return 0;
}
```

Sample Input & Output:

```
Stack elements: 30 20 10
Popped: 30
Stack elements: 20 10
```

Result: Stack operations were successfully implemented, handling overflow/underflow and validating stack correctness.