



Module 05

Partha Pratim
Das

Objectives &
Outline

Stack in C
Reverse a String
Eval Postfix

Stack in C++
Reverse a String
Eval Postfix

Summary

Module 05: Programming in C++

Stack and its Applications

Partha Pratim Das

Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

ppd@cse.iitkgp.ernet.in

Tanwi Mallick
Srijoni Majumdar
Himadri B G S Bhuyan



Module Objectives

Module 05

Partha Pratim
Das

Objectives & Outline

Stack in C

Reverse a String
Eval Postfix

Stack in C++

Reverse a String
Eval Postfix

Summary

- Understanding implementation and use of stack in C
- Understanding stack in C++ standard library and its use



Module Outline

Module 05

Partha Pratim
Das

Objectives & Outline

Stack in C

Reverse a String
Eval Postfix

Stack in C++

Reverse a String
Eval Postfix

Summary

- Stack in C
 - Reverse a String
 - Evaluate a Postfix Expression
- Stack in C++
 - Reverse a String
 - Evaluate a Postfix Expression



Understanding Stack in C

Module 05

Partha Pratim
Das

Objectives &
Outline

Stack in C

Reverse a String
Eval Postfix

Stack in C++

Reverse a String
Eval Postfix

Summary

- Stack is a LIFO (last-In-First-Out) container that can maintain a collection of arbitrary number of data items – all of the same type
- To create a stack in C we need to:
 - Decide on the data type of the elements
 - Define a structure (container) (with maximum size) for stack and declare a top variable in the structure
 - Write separate functions for push, pop, top, and isempty using the declared structure
- Note:
 - Change of the data type of elements, implies re-implementation for all the stack codes
 - Change in the structure needs changes in all functions
- Unlike `sin`, `sqrt` etc. function from C standard library, we do not have a ready-made stack that we can use



Common C programs using stack

Module 05

Partha Pratim Das

Objectives & Outline

Stack in C

Reverse a String
Eval Postfix

Stack in C++

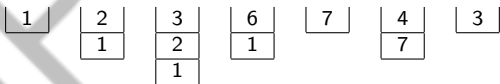
Reverse a String
Eval Postfix

Summary

Some common C programs that use stack:

- Reversing a string
 - Input: ABCDE
 - Output: EDCBA
- Evaluation of postfix expression
 - Input: $1\ 2\ 3\ * +\ 4\ -$ (for $1 + 2 * 3 - 4$)
 - Output: 3

Stack states:



- Identification of palindromes (w/ and w/o center-marker)
- Conversion of an infix expression to postfix
- Depth-first Search (DFS)



Program 05.01: Reversing a string

Module 05

Partha Pratim Das

Objectives & Outline

Stack in C
Reverse a String
Eval Postfix

Stack in C++
Reverse a String
Eval Postfix

Summary

```
// FileName: Reverse_String.c

#include <stdio.h>

typedef struct stack {
    char data [100];
    int top;
} stack;

int empty (stack *p) {
    return (p->top == -1);
}

int top (stack *p) {
    return p -> data [p->top];
}

void push (stack *p, char x) {
    p -> data [++(p -> top)] = x;
}

void pop (stack *p) {
    if (!empty(p)) {
        (p->top) = (p->top) -1;
    }
}
```

```
void main() {
    stack s;
    s.top = -1;

    char ch, str[10] = "ABCDE";

    int i, len = sizeof(str);

    for(i = 0; i < len; i++) {
        push(&s, str[i]);
    }

    printf ("Reversed String: ");

    while (!empty(&s)){
        printf("%c ", top(&s));
        pop(&s);
    }
}
```

Reversed String: EDCBA



Program 05.02: Postfix Expression Evaluation

Module 05

Partha Pratim Das

Objectives & Outline

Stack in C
Reverse a String
Eval Postfix

Stack in C++
Reverse a String
Eval Postfix

Summary

```
// FileName: PostFix_Evaluation.c
```

```
#include<stdio.h>
```

```
typedef struct stack {  
    char data [100];  
    int top;  
} stack;
```

```
int empty (stack *p) {  
    return (p->top == -1);  
}
```

```
int top (stack *p) {  
    return p -> data [p->top];  
}
```

```
void push (stack *p, char x) {  
    p -> data [++(p -> top)] = x;  
}
```

```
void pop (stack *p) {  
    if (!empty(p)) {  
        (p->top) = (p->top) -1;  
    }  
}
```

```
void main() {  
    stack s;  
    s.top = -1;  
  
    // Postfix expression: 1 2 3 * + 4 -  
    char postfix[] = {'1','2','3','*','+','4','-'};  
  
    int i, op1, op2;  
  
    for(i = 0; i < 7; i++) {  
        char ch = postfix[i];  
        if (isdigit(ch)) push(&s, ch-'0');  
        else {  
            op2 = top(&s); pop(&s);  
            op1 = top(&s); pop(&s);  
            switch (ch) {  
                case '+':push(&s, op1 + op2);break;  
                case '-':push(&s, op1 - op2);break;  
                case '*':push(&s, op1 * op2);break;  
                case '/':push(&s, op1 / op2);break;  
            }  
        }  
    }  
    printf("Evaluation %d\n", top(&s));  
}
```

Evaluation 3



Understanding Stack in C++

Module 05

Partha Pratim
Das

Objectives &
Outline

Stack in C
Reverse a String
Eval Postfix

Stack in C++
Reverse a String
Eval Postfix

Summary

- C++ standard library provide a ready-made stack for any type of elements
- To create a stack in C++ we need to:
 - Include the stack header
 - Instantiate a stack with proper element type (like `char`)
 - Use the functions of the stack objects for stack operations



Program 05.03: Reverse a String in C++

Module 05

Partha Pratim Das

Objectives & Outline

Stack in C
Reverse a String
Eval Postfix

Stack in C++
Reverse a String
Eval Postfix

Summary

```
// FileName: Reverse_String_c++.cpp
#include<iostream>
#include<string.h>
#include<stack>
using namespace std;

int main() {
    char str[10]= "ABCDE";
    stack<char> s;
    int i;

    for(i = 0; i < strlen(str); i++)
        s.push(str[i]);

    cout << "Reversed String: ";

    while (!s.empty()) {
        cout << s.top();
        s.pop();
    }

    return 0;
}
```

- No codes for creating stack
- No initialization
- Clean interface for stack functions
- Available in library – well-tested

```
// FileName: Reverse_String.c

int main() {
    char str[10]= "ABCDE";
    stack s; s.top = -1;
    int i;

    for(i = 0; i < strlen(str); i++)
        push(&s, str[i]);

    printf ("Reversed String: ");

    while (!empty(&s)){
        printf ("%c ", top(&s));
        pop(&s);
    }

    return 0;
}
```

- Lot of code for creating stack
- top to be initialized
- Cluttered interface for stack functions
- Implemented by user – error-prone



Program 05.04: Postfix Evaluation in C++

Module 05

Partha Pratim
Das

Objectives &
Outline

Stack in C

Reverse a String
Eval Postfix

Stack in C++

Reverse a String
Eval Postfix

Summary

```
// FileName:Postfix_Evaluation_c++.cpp
#include <iostream>
#include <stack>
using namespace std;

int main() {
    // Postfix expression: 1 2 3 * + 4 -
    char postfix[] = {'1','2','3','*','+','4','-'}, ch;
    stack<int> s;

    for(int i = 0; i < 7; i++) {
        ch = postfix[i];
        if (isdigit(ch)) { s.push(ch-'0'); }
        else {
            int op1 = s.top(); s.pop();
            int op2 = s.top(); s.pop();
            switch(ch) {
                case '*': s.push(op2 * op1); break;
                case '/': s.push(op2 / op1); break;
                case '+': s.push(op2 + op1); break;
                case '-': s.push(op2 - op1); break;
            }
        }
    }
    cout << "\nEvaluation " << s.top();
    return 0;
}
```



Module Summary

Module 05

Partha Pratim
Das

Objectives &
Outline

Stack in C
Reverse a String
Eval Postfix

Stack in C++
Reverse a String
Eval Postfix

Summary

- C++ standard library provides ready-made stack. It works like a data type
- Any type of element can be used for C++ stack
- Similar containers as available in C++ standard library include:
 - queue
 - deque
 - list
 - map
 - set
 - ... and more



Instructor and TAs

Module 05

Partha Pratim
Das

Objectives &
Outline

Stack in C
Reverse a String
Eval Postfix

Stack in C++
Reverse a String
Eval Postfix

Summary

Name	Mail	Mobile
Partha Pratim Das, <i>Instructor</i>	ppd@cse.iitkgp.ernet.in	9830030880
Tanwi Mallick, <i>TA</i>	tanwimallick@gmail.com	9674277774
Srijoni Majumdar, <i>TA</i>	majumdarsrijoni@gmail.com	9674474267
Himadri B G S Bhuyan, <i>TA</i>	himadribhuyan@gmail.com	9438911655