UCSC Silicon Valley Extension

Advanced C Programming

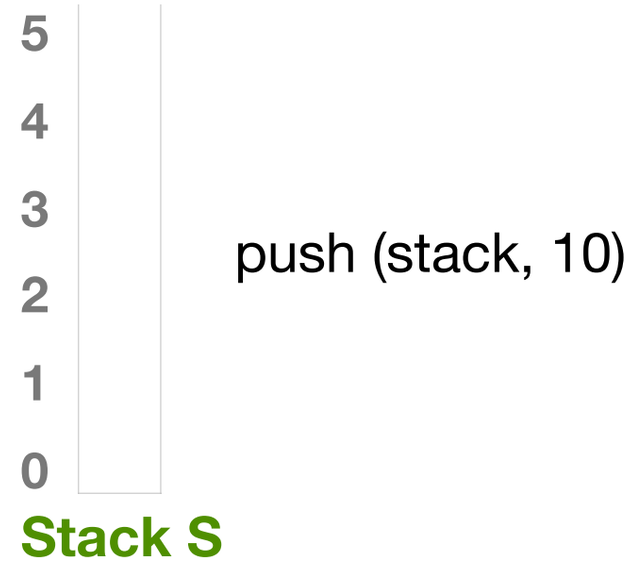
Stacks

Radhika Grover

Overview

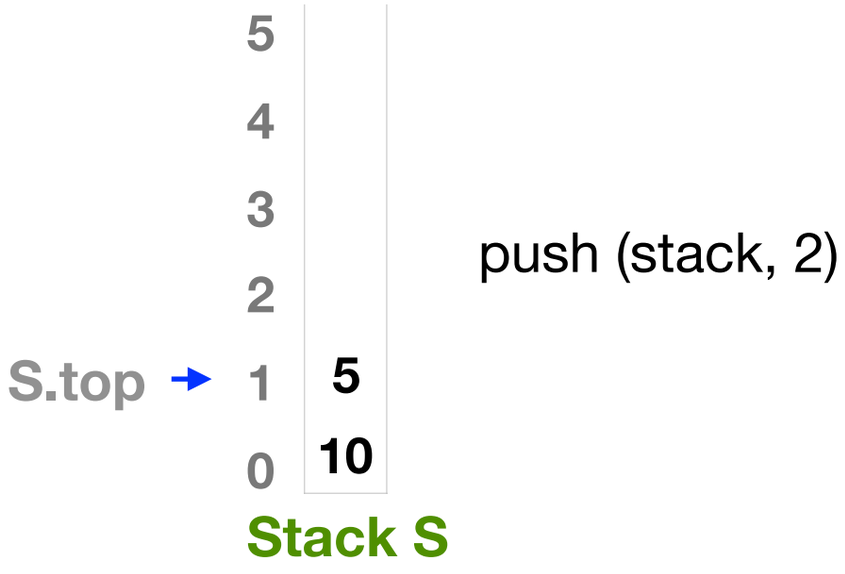
- Stacks
- Example
- Implementation using an array
- Applications

Example of stack

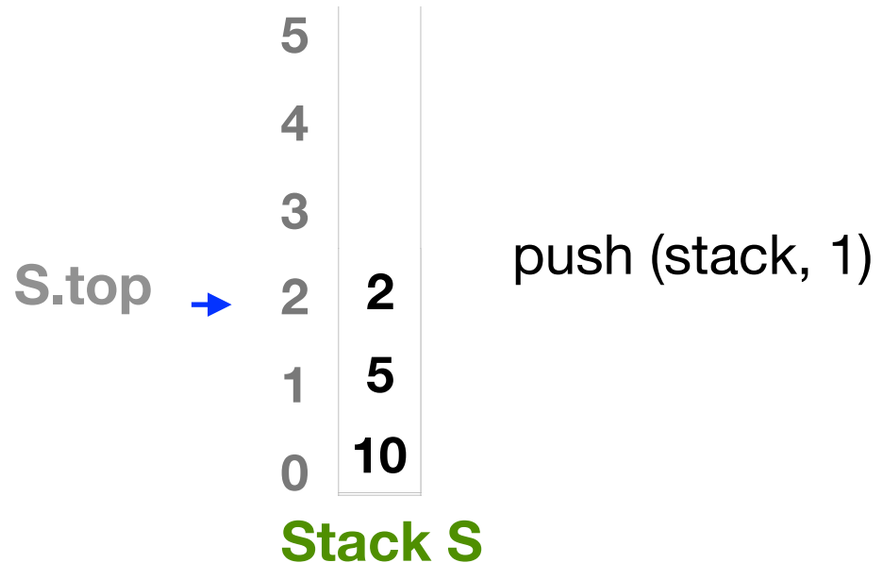


Example of stack

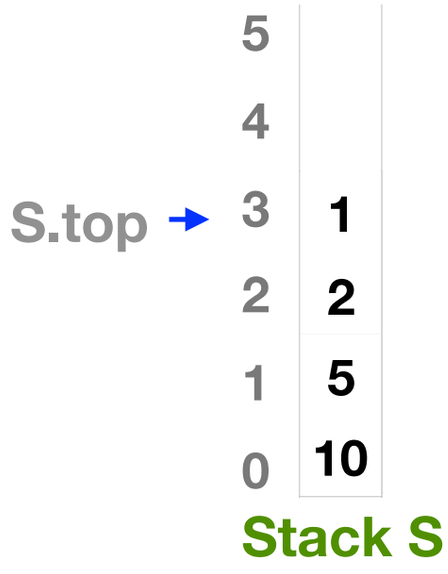
Example of stack



Example of stack

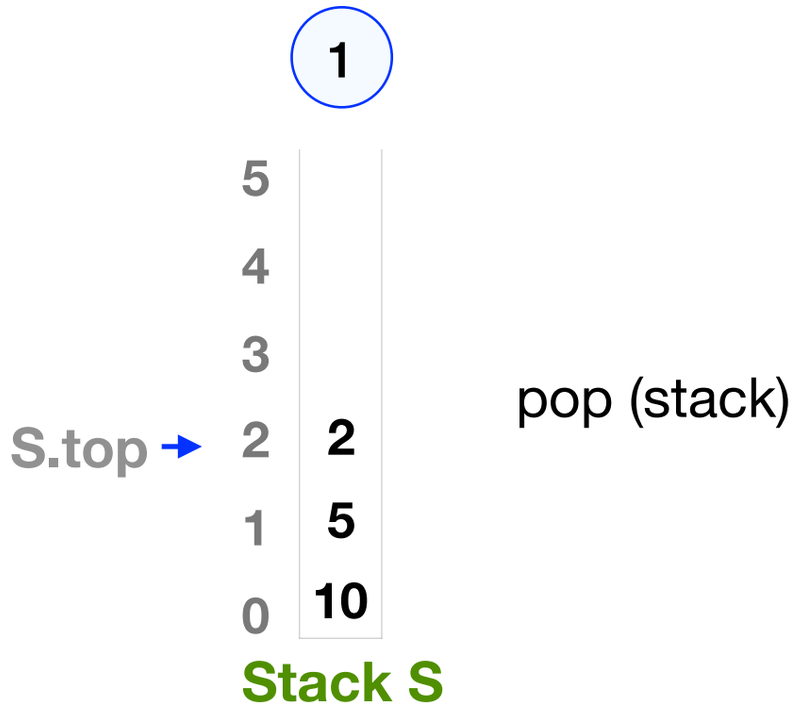


Example of stack

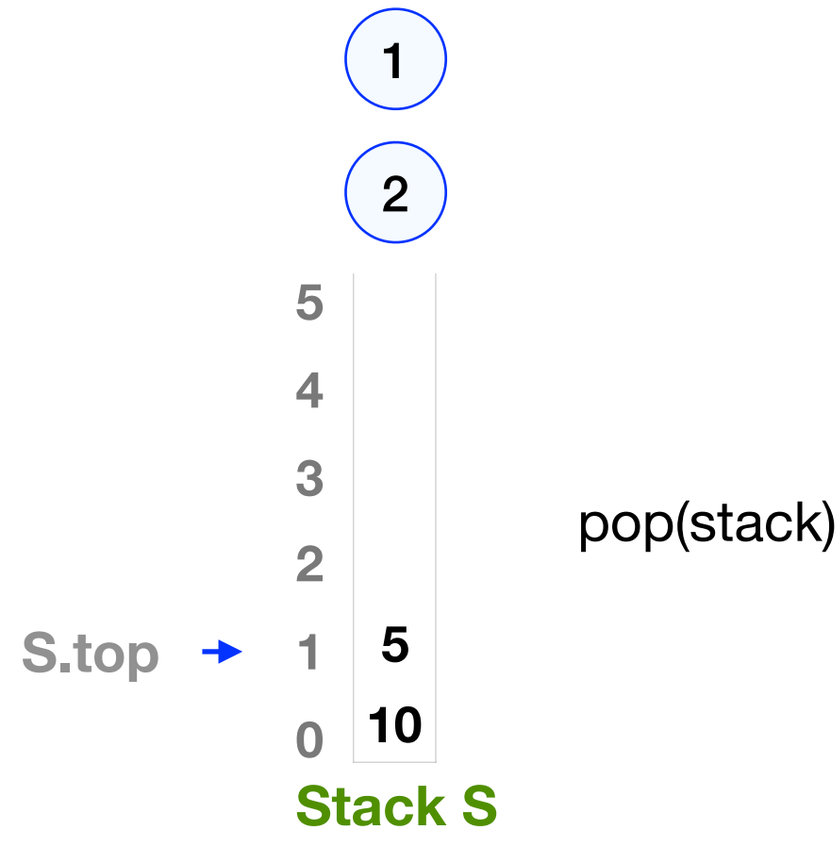


pop (stack)

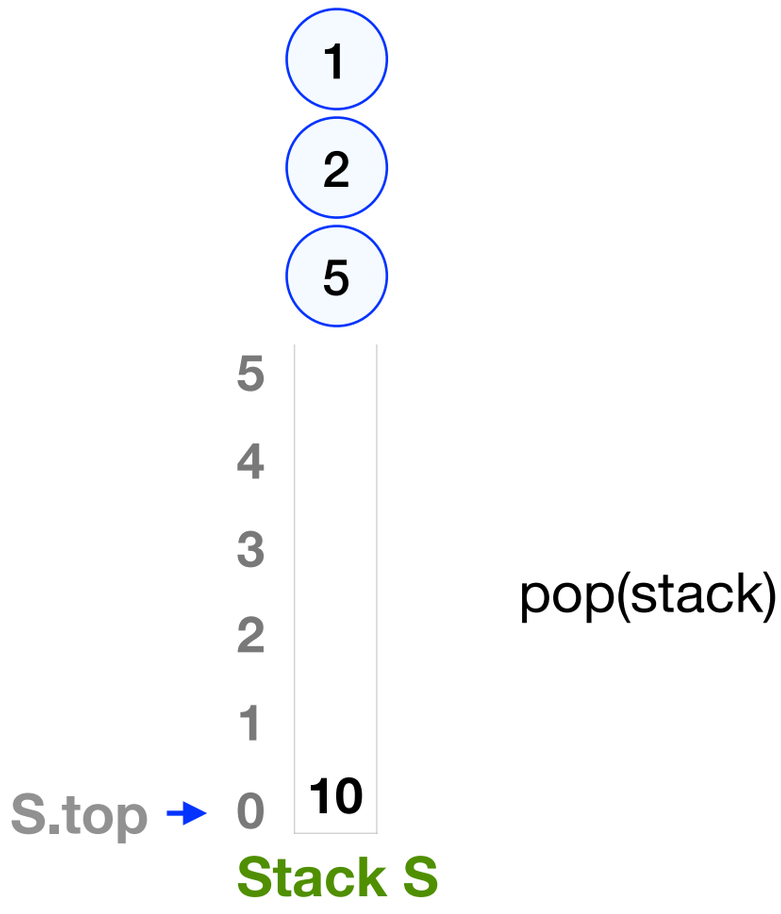
Example of stack



Example of stack

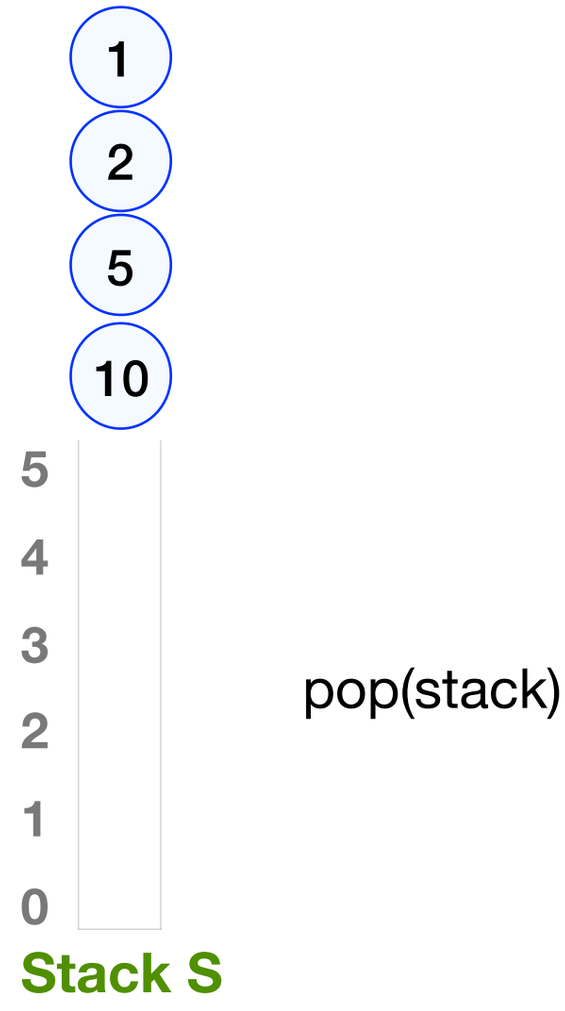


Example of stack



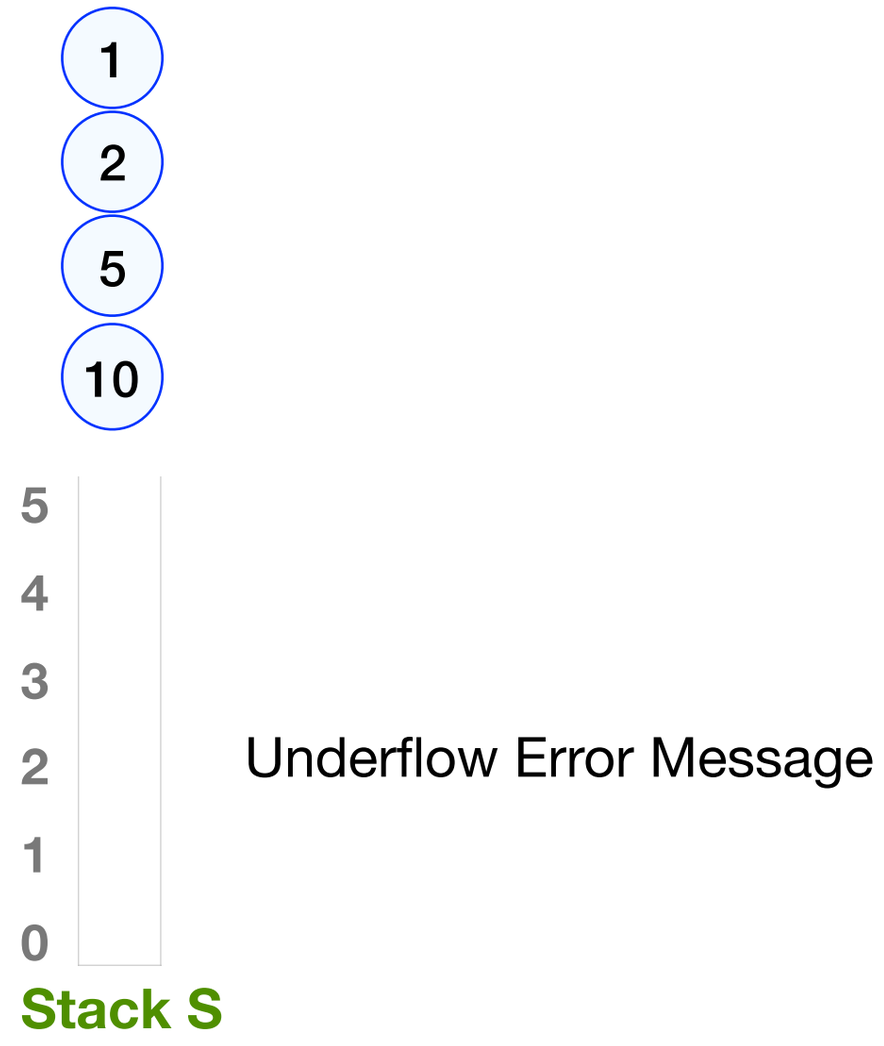
pop(stack)

Example of stack



pop(stack)

Example of stack



Stack pseudocode

Array based implementation:

```
push(S, value){
    if (isStackFull())
        Overflow Error;
    S.top = S.top + 1;
    S[top] = value;
}
```

```
pop(S){
      if (isStackEmpty()){
         Underflow Error;
      }
      S.top = S.top - 1;
}
```

```
isStackEmpty(S){
   if (S.top == -1)
       return true;

   return false;
}
```

```
top(S){
      if (isStackEmpty()){
         Underflow Error;
      }
      return S.top;
}
```

Time complexity = O(1)

Stack implementation using an array

```
//Stack/stack.c
#define MAX_SIZE  100

typedef struct data_t {
    int value;
} Data;

typedef struct Stack_t {
    int topIndex;
    int capacity;
    Data *data;
} Stack;
```

Stack implementation using an array

```
//Stack/stack.c
// push newData on stack
void push(Stack *stack, Data newData) {
    if (stack->topIndex == stack->capacity) {
        fprintf(stderr, "ERROR: Stack is filled to capacity");
        exit(EXIT_FAILURE);
    }
    stack->topIndex++;
    stack->data[stack->topIndex] = newData;
}

// pop stack (decrement stack pointer)
void pop(Stack *stack) {
    if (stack->topIndex == -1) {
        fprintf(stderr, "Stack is empty");
        exit(EXIT_FAILURE);
    }
    stack->topIndex--;
}
```

Stack implementation using an array

```
//Stack/stack.c
// return topmost item of stack
Data top(Stack *stack) {
    if (stack->topIndex == -1) {
        fprintf(stderr, "Stack is empty”);
        exit(EXIT_FAILURE);
    }

    return(stack->data[stack->topIndex]);
}

bool isEmpty(Stack *stack) {
    return (stack->topIndex == -1);
}
```

Check for memory leaks

- Use a tool to check for memory leaks.

- For example, for Linux systems, you can run valgrind on the command line for the executable Stack as:

- valgrind --tool=memcheck --leak-check=full --show-leak-kinds=all ./ Stack

Applications

- Used in function calls for storing arguments, temporary variables and return type while the function executes.
- Evaluating expressions
- During Depth first search
- The Java Virtual Machine has a stack-based architecture.