

Write a C program to implement the application of Stack (Notations)

```
// Stack implementation in C
```

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

```
#include <stdlib.h>
```

```
#define MAX 10
```

```
int count = 0;
```

```
// Creating a stack
```

```
struct stack {
```

```
    int items[MAX];
```

```
    int top;
```

```
};
```

```
typedef struct stack st;
```

```
void createEmptyStack(st *s) {
```

```
    s->top = -1;
```

```
}
```

```
// Check if the stack is full
```

```
int isfull(st *s) {
```

```
    if (s->top == MAX - 1)
```

```
        return 1;
```

```
    else
```

```
        return 0;
```

```
}
```

```
// Check if the stack is empty
```

```
int isempty(st *s) {  
    if (s->top == -1)  
        return 1;  
    else  
        return 0;  
}
```

// Add elements into stack

```
void push(st *s, int newitem) {  
    if (isfull(s)) {  
        printf("STACK FULL");  
    } else {  
        s->top++;  
        s->items[s->top] = newitem;  
    }  
    count++;  
}
```

// Remove element from stack

```
void pop(st *s) {  
    if (isempty(s)) {  
        printf("\n STACK EMPTY \n");  
    } else {  
        printf("Item popped= %d", s->items[s->top]);  
        s->top--;  
    }  
    count--;  
    printf("\n");  
}
```

// Print elements of stack

```
void printStack(st *s) {  
    printf("Stack: ");  
    for (int i = 0; i < count; i++) {  
        printf("%d ", s->items[i]);  
    }  
    printf("\n");  
}
```

```
// Driver code
```

```
int main() {  
    int ch;  
    st *s = (st *)malloc(sizeof(st));  
  
    createEmptyStack(s);  
  
    push(s, 1);  
    push(s, 2);  
    push(s, 3);  
    push(s, 4);  
  
    printStack(s);  
  
    pop(s);  
  
    printf("\nAfter popping out\n");  
    printStack(s);  
}
```