

8/1/2024

USN-1BM22CS241 (Sanjeet P. Pandit)

CODE:

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

```
#include <stdbool.h>
```

```
bool isValid(char* s) {
```

```
    int top = -1;
```

```
    char stack[1000];
```

```
    while (*s != '\0') {
```

```
        if (*s == '(' || *s == '{' || *s == '[') {
```

```
            stack[++top] = *s;
```

```
        } else {
```

```
            if (top == -1)
```

```
                return false;
```

```
            char topElement = stack[top--];
```

```
            if ((*s == ')' && topElement != '(') ||
```

```
                (*s == '}' && topElement != '{') ||
```

```
                (*s == ']' && topElement != '[')) {
```

```
                return false;
```

```
            }
```

```

    }

    s++;

}

return top == -1;

}

int main() {

    char input[1000];

    printf("Enter a string: ");

    scanf("%s", input);

    printf("%s\n", isValid(input) ? "Valid" : "Invalid");

    return 0;

}

```

OUTPUT:

Enter a String: { ([]) }

Valid

Enter a String: (}]

Invalid

20. Valid Parentheses

Attempted 🟡

Easy Topics Companies Hint

Given a string `s` containing just the characters `'('`, `')'`, `'{'`, `'}'`, `'['` and `']'`, determine if the input string is valid.

An input string is valid if:

1. Open brackets must be closed by the same type of brackets.
2. Open brackets must be closed in the correct order.
3. Every close bracket has a corresponding open bracket of the same type.

Example 1:

Input: `s = "()"`

Output: `true`

Example 2:

Input: `s = "()[]{}"`

Output: `true`

Example 3:

Input: `s = "{}"`

Output: `false`

Constraints:

- $1 \leq s.length \leq 10^4$
- `s` consists of parentheses only `'()[]{}'`.

```
1 #include <stdio.h>
2 #include <stdbool.h>
3
4 bool isValid(char* s) {
5     int top = -1;
6     char stack[1000];
7
8     while (*s != '\0') {
9         if (*s == '(' || *s == '{' || *s == '[') {
10             stack[++top] = *s;
11         } else {
12             if (top == -1)
13                 return false;
14
15             char topElement = stack[top--];
16
17             if ((*s == ')' && topElement != '(') ||
18                 (*s == '}' && topElement != '{') ||
19                 (*s == ']' && topElement != '[')) {
20                 return false;
21             }
22         }
23         s++;
24     }
25     return top == -1;
26 }
27 }
```

Saved to local

Ln 14, Col 17

Accepted Runtime: 0 ms

Case 1 Case 2 Case 3