# Stack

## Filling-up blank

Read instructions, if it is a char, push to stack, otherwise pop out and print

- input: `EAS*Y*QUE***ST***IO*N***`
- output: `SYEUQTSAONIE`

```
#include <bits/stdc++.h>
using namespace std;
const int N = 100;
int s[N], idx;

int main() {

  char str[] = "EAS*Y*QUE***ST***IO*N***";
  // init stack
  idx = ?;

  for (int i=0; i<strlen(str); i++) {
    if (str[i] != '*') {
      ?
    }
    else {
      printf("%c", s[?]);
    }
  }
  printf("\n");
  return 0;
}
```

## Brackets matching

Give a string with brackets, print indexes of each matched bracket,
e.g.:

- For `(())`

  ```
  1 2
  0 3
  ```

- For `()()`

  ```
  0 1
  2 3
  ```

## Monotonic stack

Give prices sorted by quality (from low to high),
print all candidates when we consider first $i$ items,
e.g.:

- For `{4, 6, 2, 7}`

  ```
  4
  4 6
  2
  2 7
  ```

# Queue

## Filling-up blank

Read instructions, if it is a char, push to queue, otherwise pop out and print

```
#include <bits/stdc++.h>
using namespace std;
const int N = 100;
int que[N], front, tail;

int main() {
  char str[] = "EAS*Y*QUE***ST***IO*N***";
  // init queue
  ?

  for (int i=0; i<strlen(str); i++) {
    if (str[i] != '*') {
      ?
    }
    else {
      printf("%c", ?);
    }
  }
  printf("\n");
  return 0;
}
```

## Range max (monotonic queue)

Given $n$ integers $a_1, a_2, \ldots, a_n$, and an integer $k$, find $max(a_i, a_{i+1}, \ldots, a_{i+k})$ for all $1 \leq i < n$.

- e.g.: for `k=2, a = {1,3,2,5,4}`
  - output:

    ```
    3
    3
    5
    5
    ```

# Common mistakes

## Integer overflow

```
#include <bits/stdc++.h>
using namespace std;

int floorsqrt(int n) {
  int i = 1;
  while (i * i < n) i++;
  return i;
}

int main() {
  cout << floorsqrt(2147413644) << endl;
  cout << floorsqrt(2147483647) << endl;
}
```