C++ Basics (Part 2)

Goal

To understand:

- Loops
- Arrays
- Functions
- Range of datatypes
- Headers and Namespaces
- Competitive Programming tricks
- Basic C++ Template for CP

Loop

Loops are used to repeat a block of code until some condition is satisfied.

There are three types of loops in C++:

- for loop
- while loop
- do-while loop

Loop (Miscellaneous)

- An iteration is defined as one time the loop gets executed. For example, 3rd iteration is the 3rd time the loop is run.
- “break” statement exits the current/innermost loop when executed.
- “continue” statement skips to the next iteration of the current/innermost loop when executed.

“for” loop

Syntax:

```
for (s1; s2; s3) {
    // Code here
}
```

s1: Executed once before start of loop.

s2: Condition of the loop. Loop exits if false.

s3: Executed after each iteration.

“while” loop

Syntax:

```
while (     ) {
    // Code here
}
```

Check if the condition is true and then execute the block of code. Repeat.

Arrays

An array is a collection of multiple items of the same datatype.

- Arrays are ordered.
- The size of an array cannot be changed.

Syntax: `datatype name[size]`

Functions

Functions are reusable blocks of code that can be run whenever called.

They can take in parameters (input) and return a value (output).

Syntax:

```
return_type name(d1 param1, d2 param2, …) {
    // result must be same as return_type
    return result;
}
```

Range of integer types

- `int`: (-2^{31}) to $(2^{31} - 1)$
  `2^31` is a bit higher than $2*10^9$
- `long int`: Almost always same as int
- `long long int`: (-2^{63}) to $(2^{63} - 1)$
  `2^63` is a bit higher than $9*10^{18}$
- `float/double/long double`:
  `7` digits / `15` digits / `18` digits

NOTE: Make sure you don't print big numbers with floating-point datatypes

Namespace

A namespace is a scope of the program that can store various useful functions and variables.

Two ways to use namespaces:

- Use scope resolution operator "::" (double colon) to use the values inside the namespace
- Type `using namespace name;` at the start of the file.

Namespaces are used to avoid conflicting names.

Header files

Header files store C++ variables, functions, etc. to be shared with multiple files

- Pre-existing header files:
  Files provided by the compiler for a variety of purposes.

- User-defined header files: Files written by the user.
  Can be used for templates, or to make code less complex.

Syntax:

```
#include <filename>
```

Header file for Competitive Programming

```
#include <bits/stdc++.h>
```

Pros:

- Includes every standard library and STL headers. Therefore, it saves time spent coding during contests.

Cons:

- Increases compile time (Doesn’t matter for CP)
- Does not work with compilers other than GNU C++

Fast I/O

```
ios::sync_with_stdio(false);
```

Removes sync between cout and printf.

```
cin.tie(NULL);
```

Removes sync between cout and cin.

```
endl vs '\n'
```

"endl" forces the input buffer to flush.
When using fastio, use '\n' rather than endl

https://www.youtube.com/watch?v=aNF4DEluWnI
(my video on fast I/O)

My template:

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

#define endl '\n'
#define int long long

const int MOD = 1e9 + 7;
const int INF = LLONG_MAX >> 1;

signed main() {
    ios::sync_with_stdio(false); cin.tie(NULL);

    int tc; cin >> tc;
    while (tc--) {

    }
}
```

Problems:

Array problems:

- https://codeforces.com/problemset/problem/110/A
- https://cses.fi/problemset/task/1083
- https://codeforces.com/problemset/problem/677/A

String problems:

- https://cses.fi/problemset/task/1069
- https://codeforces.com/problemset/problem/1619/A

Resources:

- https://www.youtube.com/watch?v=aNF4DEluWnI (my video on fast I/O)
- https://www.tutorialspoint.com/cplusplus/cpp_namespaces.htm (for namespaces)
- https://www.programiz.com/cpp-programming/multidimensional-arrays (multi-dimensional arrays)
- https://devdocs.io/ (docs for all in-built features)

Thanks for watching!