Programming Seminar

XXIX ACIS Colombian Programming Contest tips

Santiago Vanegas Gil
Esteban Foronda Sierra

EAFIT University

Monday, August 31st 2015

Content

Set up

Get ready

Open your favourite editor.

We recommend Dev C++ or CodeBlocks for C/C++

NetBeans or Eclipse for Java

Content

What is a template?

The template is the code contained by every program in the contest, i.e. a base code.
For example:

Main method

The main method should be included in all programs.

```
int
main() {
  // Your code goes here.
  return 0;
}
```

Let's write a template

Let's imagine that we already are in the contest. How would the editor be configured?

Let's write a template

Let's imagine that we already are in the contest. How would the editor be configured?

Test 1

5 minutes

Let's write a template

Let's imagine that we already are in the contest. How would the editor be configured?

Test 1

5 minutes

Each team should write a template for their favourite programming language. *(We hope it is C++)*

The first team writting a good template and configuring the editor will win.

A fast and enough template

- Did you include lots of header files?
- Did you write `using namespace std;`?
- Did you write the main method?

A fast and enough template

- Did you include lots of header files?
- Did you write `using namespace std;`?
- Did you write the main method?

Template

```
#include <bits/stdc++.h> // Just include this header.

using namespace std; // Don't forget this.

int
main() {
  // Your code goes here.
  return 0;
}
```

Content

Common types of input

In a programming contest, the most common input method is the standard one, i.e. reading from **standard input**. However, you could be required to **read from a file**, let's see how to do it.

Reading from a file in C++

You can use either input and output file streams (`ifstream`, `ofstream`) or just redirect the standard input and output to a file.

Reading from a file in C++

You can use either input and output file streams (`ifstream`, `ofstream`) or just redirect the standard input and output to a file.

Redirect standard streams

```
freopen("file.in", "r", stdin);
freopen("file.out", "w", stdout);
int num;
cin >> num;
cout << "I read: " << num << endl;
```

Reading from a file in C++

You can use either input and output file streams (`ifstream`, `ofstream`) or just redirect the standard input and output to a file.

Redirect standard streams

```
freopen("file.in", "r", stdin);
freopen("file.out", "w", stdout);
int num;
cin >> num;
cout << "I read: " << num << endl;
```

Use new streams

```
ifstream fin("file.in");
ofstream fout("file.out");
int num;
fin >> num;
fout << "I read: " << num << endl;
```

Reading from a file in Java

You should use the `FileInputStream` class to read from a file.

Reading from a file in Java

You should use the `FileInputStream` class to read from a file.

Redirect standard streams

```
import java.io.*;
import java.util.*;

class Main {
  public static void main(String [] args)
    throws FileNotFoundException {

    System.setIn(new FileInputStream("file.in"));
    System.setOut(new PrintStream("file.out"));

    Scanner in = new Scanner(System.in);
    int n = in.nextInt(); String a = in.next();
  }
}
```

Let's read some inputs

Use any reading method.

Let's read some inputs

Use any reading method.

Test 2

4 minutes

Let's read some inputs

Use any reading method.

Test 2

4 minutes

Each team shoud write a program that reads from standard input an arbitrary set of integers and strings.
You don't know how many elements are in the input, you only have to print each one in a single line.
An element is a number or a string that is separated from another element by at least one space or end of line.

Sample input

```
hi 251
mornin6a
read 1 this
```

Sample output

```
hi
251
mornin6a
read
1
this
```

Reading from standard input

You can use `cin`, `scanf`, `getline`. Depending on your needs. Take care of:

- Usage of `cin` with `getline`: *use cin.ignore()*
- Usage of `cin` with `scanf`: *write* `ios::sync_with_stdio(false);` *in the first line of your main method when using only* `cin` *to get a better performance.*

For Java, use `Scanner` or `BufferedReader`.

Reading from standard input in C++

For the previous test we could use `cin`.

Test2 solution

```
int
main() {
  string s;
  while (cin >> s) cout << s << endl;
}
```

Reading from standard input in C++

For the previous test we could use `cin`, with no `stdio` syncing.

Test2 solution

```
int
main() {
  // When using only cin and cout, write this to
  // get a better performance.
  ios::sync_with_stdio(false);
  string s;
  while (cin >> s) cout << s << endl;
}
```

Scanf reading tricks

Scanf tricks.

Test 3

5 minutes

Scanf reading tricks

Scanf tricks.

Test 3

5 minutes

Write a program that reads multiple lines, each line will contain two colon and semicolon separated integers. Output one line for each pair of integers, using right justified width of 5, and leading zeros. Both integers shoud be separated by a '-'character.

Sample input	**Sample output**
`56:;14`	`00056 - 00014`
`99999:;1`	`99999 - 00001`
`1:;0`	`00001 - 00000`

Scanf reading tricks

Solution

Test 3 solution

```
int
main() {
  int a, b;
  while (scanf("%d:;%d", &a, &b) != EOF) // EOF!
    printf("%05d - %05d\n", a, b);
  return 0;
}
```

Reading tricks

We are asked to read multiple lines, each one with an arbitrary number of integers.

Reading tricks

We are asked to read multiple lines, each one with an arbitrary number of integers.

Test 4

5 minutes

Reading tricks

We are asked to read multiple lines, each one with an arbitrary number of integers.

Test 4

5 minutes

Write a program that reads several lines, each line will contain an unknown number of integers. For each line, print the sum of all integers in that line.

Sample input	**Sample output**
1 2 3	6
	0
5 5 5 5 5	25

Stringstream tricks

Solution using `stringstream`.

Test 4 solution

```
int
main() {
  string line;
  while (getline(cin, line)) {
    int current, sum = 0;
    stringstream ss(line);
    while (ss >> current) sum += current;
    cout << sum << endl;
  }
  return 0;
}
```

Reading tricks

Now we are asked to solve the previous problem, but they will give us the number of lines following in the input.

Reading tricks

Now we are asked to solve the previous problem, but they will give us the number of lines following in the input.

Test 5

3 minutes

Reading tricks

Now we are asked to solve the previous problem, but they will give us the number of lines following in the input.

Test 5

3 minutes

Write a program that reads an integer, inditating the number of lines to follow, each line will contain an unknown number of integers. For each line, print the sum of all integers in that line.

Sample input	**Sample output**
3	6
1 2 3	0
	25
5 5 5 5 5	

Stringstream tricks

Solution using `stringstream`.

Test 5 solution

```
int
main() {
  int n; cin >> n;
  cin.ignore(); // EXTREMELY IMPORTANT.
  for (int i = 0; i < n; ++i) {
    string line; getline(cin, line);
    int current, sum = 0;
    stringstream ss(line);
    while (ss >> current) sum += current;
    cout << sum << endl;
  }
  return 0;
}
```

Content

Operations in numbers

Suppose that we have the following problem:

Test 6

3 minutes

Operations in numbers

Suppose that we have the following problem:

Test 6

3 minutes

You are given two integers, in range $[1, 2^{30}]$, you have to add them and output the result.

Operations in numbers

As $2 * 2^{30}$ doesn't fit in an 32-bit integer, we should use 64-bit integers.
Use `long long` for C++ or `long` for Java.

Operations in numbers

As $2 * 2^{30}$ doesn't fit in an 32-bit integer, we should use 64-bit integers.
Use `long long` for C++ or `long` for Java.

Test 6 solution

```
long long a, b;
cin >> a >> b;
cout << a + b << endl;
```

Operations in numbers

As $2 * 2^{30}$ doesn't fit in an 32-bit integer, we should use 64-bit integers.
Use `long long` for C++ or `long` for Java.

Test 6 solution

```
long long a, b;
cin >> a >> b;
cout << a + b << endl;
```

You can define an alias to avoid writing `long long` in the whole code.

```
typedef long long ll;

ll a, b;
cin >> a >> b;
cout << a + b << endl;
```

Operations in numbers

What if the input numbers doesn't even fit in a 64-bit integer?

Operations in numbers

What if the input numbers doesn't even fit in a 64-bit integer? We should use the `BigInteger` Java class. Or build our own big numbers operations in C++.

Java BigInteger

```
// DO NOT EVER INCLUDE A PACKAGE.

import java.util.*;
import java.math.BigInteger;

// BE CAREFUL WITH THE CLASS NAME.
// It should be the same as the problem basename in BOCA Problems tab.
public class biginteger {
  public static void main (String [] args) {
    Scanner in = new Scanner(System.in);
    BigInteger a = in.nextBigInteger();
    BigInteger b = in.nextBigInteger();
    System.out.println(a.add(b));
  }
}
```

Content

Thanks!

We hope to see all of us advancing to the next round.
ICPC Latin America-North Regionals at Bogotá.