

Contents

1	Virtual Judge	2
1.1	Registration	2
1.2	Locating the Contests	3
1.3	Navigating the Problem Set	4
1.4	Submitting Your Solution	5
2	Solving a Problem	6
2.1	Reading the Problem Statement	6
2.2	Writing Code and Testing	7
2.2.1	Basics	7
2.2.2	Exit Code of Your Program	7
2.2.3	Starter Code for C++, Java and Python	7
2.3	Submitting Your Solution	7
3	Understanding The Judge's Verdicts	8
4	Working With Command Line Prompt	9
4.1	Compile and Run Your Code	9
4.1.1	C++	9
4.1.2	Java	9
4.1.3	Python	9
4.2	Input/Output Redirection	10
4.3	Other Useful Commands	10

1 Virtual Judge

Virtual judge is the platform we use to host the IPL contests.

1.1 Registration

You need to have a vjudge account in order to participate in the IPL contests. Go to vjudge.net and click "Register" in the top-right corner.

Register

* Username:

* Password:

* Repeat Password:


Nickname:

School:

QQ:

* Email:

* Captcha:



Introduction:

in Markdown

Cancel

Register

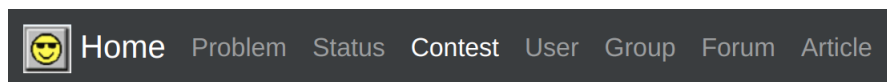
It is preferred that you use your netid as the username or the nickname. You can leave the optional questions (School, QQ, Introduction) blank.

Also remember to fill out the google form (link available on our website) that asks for your netid and vjudge id. If you do not fill out the form, there is no guarantee that we record your scores properly.

Page 2

1.2 Locating the Contests

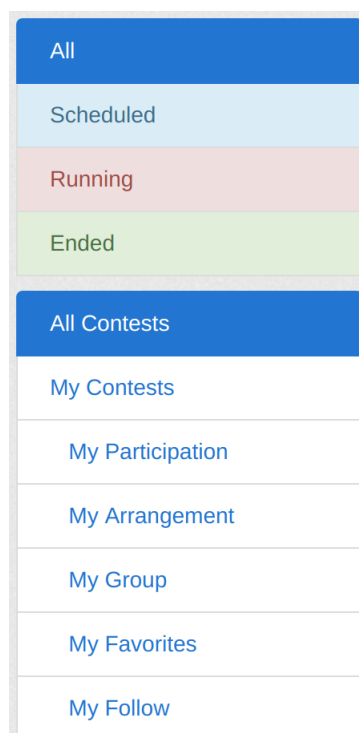
After you have logged into your account, go to "Contest" on the navigation bar.



Search for the title "IPL S2" or owner "uiuc_icpc".

ID		Title		Begin Time	Length	Owner
		<input type="text" value="IPL S2"/>				<input type="text" value="uiuc_icpc"/>
185825	☆	IPL S2 - Freshman Contest 1		2 hr later	1.7 hours	uiuc_icpc
185818	☆	IPL S2 - Open Contest 1		2 hr later	1.7 hours	uiuc_icpc
183940	☆	IPL S2 - Practice Contest	👤 x37	7 days ago	1.7 hours	uiuc_icpc

If you do not see the contests, check the side panel and set the filters to "All" and "All Contests".



Click the contest you are going to attend and it will ask you for a password, which will be announced before the contest starts.

Begin: 2017-09-18 19:20 EST

IPL S2 - Freshman Contest 1

End: 2017-09-18 21:00 EST

Start Countdown: 01:59:09

Password:

Login

1.3 Navigating the Problem Set

Once the contest has started, you will be able to see the following sections.

Overview	Problem	Status	Rank (01:40:00)	0 Comments
	Stat	#	Origin	Title
	35 / 37	A	POJ 1000	A+B Problem
	35 / 40	B	CodeChef START01	Number Mirror
	34 / 55	C	SPOJ TEST	Life, the Universe, and Everything
	27 / 80	D	CodeChef HS08TEST	ATM
	23 / 61	E	CodeChef CIELAB	Ciel and A-B Problem

- **Overview:** an overview of the problem set
- **Problem:** view and submit each of the problems

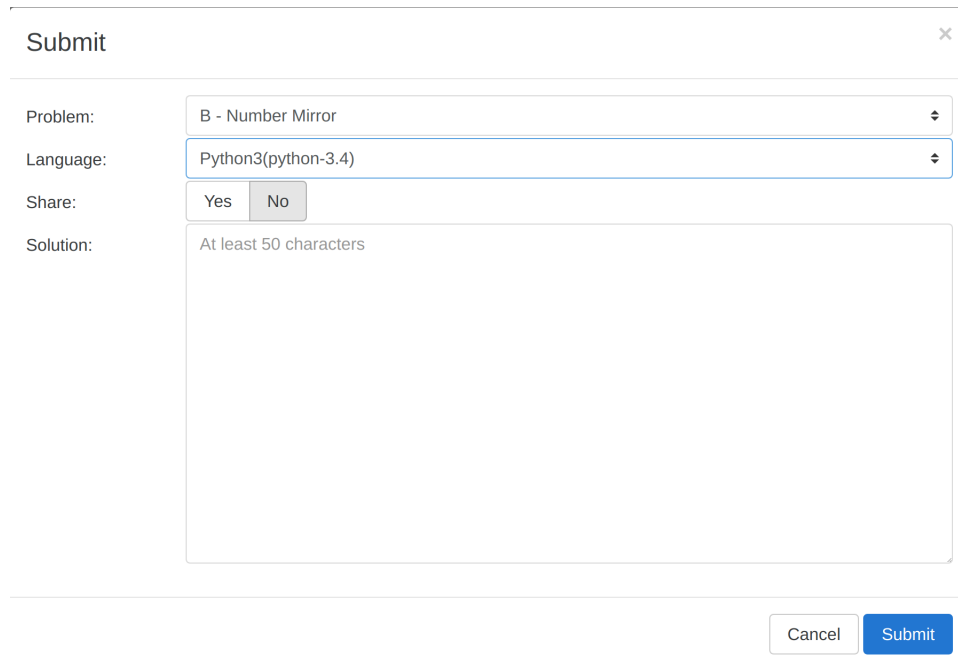
The screenshot shows the 'Problem' tab selected in the contest interface. The problem is titled 'B - Number Mirror' and is from 'CodeChef - START01'. On the left, there is a sidebar with a 'Submit' button, 'Status' and 'My Status' tabs, and a list of contest parameters: Time limit (1000 ms), Code length Limit (50000 B), OS (Linux), Language limit (C99 strict, CPP 6.3, JAVA, PYTHON, PYTHON 3.5), Author (admin), and Tags (admin). The main content area displays the 'Problem Statement' which asks to write a program that accepts a number n and outputs the same. It also shows the 'Input' (a single integer), 'Output' (a single line), and 'Constraints' ($0 \leq n \leq 10^5$). 'Sample Input' is 123 and 'Sample Output' is 123.

- **Status:** all recent submissions in this contest
- **Rank:** the current standings where you can see how other people are doing
- **Comments:** a place for you to ask questions, but we DO NOT use that (raise your hands and ask us in person!)

The problems are sorted by their difficulty levels (from easy to hard), however you are allowed to solve them in any order as you wish.

1.4 Submitting Your Solution

In the problem tab, click "Submit" and you will see this window popped up:

A modal window titled "Submit" with a close button (X) in the top right corner. It contains four rows of input fields: "Problem:" with a dropdown menu showing "B - Number Mirror"; "Language:" with a dropdown menu showing "Python3(python-3.4)"; "Share:" with two buttons, "Yes" and "No", where "No" is currently selected; and "Solution:" with a large text area containing the placeholder text "At least 50 characters". At the bottom right of the window are two buttons: "Cancel" and "Submit".

Submit

Problem: B - Number Mirror

Language: Python3(python-3.4)

Share: Yes No

Solution: At least 50 characters

Cancel Submit

Copy and paste your code to the textbox and then click "Submit".
Make sure you pick the right problem and the right programming language!

2 Solving a Problem

In the IPL (and many other programming contests), solving a problem takes the following three steps:

1. Read the Problem Statement
2. Write Code and Test
3. Submit Your Code to the Judge

2.1 Reading the Problem Statement

In most programming contests, the problem statement usually consists of several parts:

<p>A particle has initial velocity and acceleration. If its velocity after certain time is v then what will its displacement be in twice of that time?</p>	<p>Problem Description: tells you what to do; might contain background stories</p>
<p>Input</p> <p>The input will contain two integers in each line. Each line makes one set of input. These two integers denote the value of v ($-100 \leq v \leq 100$) and t ($0 \leq t \leq 200$) (t means at the time the particle gains that velocity)</p>	<p>Input Specification: Format, Meaning & Range</p>
<p>Output</p> <p>For each line of input print a single integer in one line denoting the displacement in double of that time.</p>	<p>Output Specification: how you should print the results out</p>
<p>Sample Input</p> <pre>0 0 5 12</pre> <p>Sample Output</p> <pre>0 120</pre>	<p>Sample Input & Output: one or more groups of (weak) testing data that helps you understand the problem and debug your code</p>

In addition, a problem might also have a time limit and a memory limit your program can reach at maximum:

<p>F. To Play or not to Play</p> <p>time limit per test: 2 seconds</p> <p>memory limit per test: 256 megabytes</p>

In conclusion, you need to read the problem statement, extract useful information, understand and model the problem it asks you to solve, and think of the appropriate algorithms and data structures you will use later.

It is also extremely important to pay attention to the range of the input data, as different ranges may indicate completely different algorithms and difficulty levels for the problem.

2.2 Writing Code and Testing

2.2.1 Basics

You write your code, compile it (for compiled languages), and test it on your local machine.

All your code should be contained in a single source file, no matter what language you use.

Your program should read the data from standard input, solve the problem specified in the statement, and print the results to standard output (unless otherwise specified).

Make sure you test your solution against the sample inputs and outputs, as it is a bare minimum requirement for your solution to get accepted by the judge.

2.2.2 Exit Code of Your Program

No matter what language you use, your program should always have zero as the exit code.

A non-zero exit code will make the judge think your program terminated in an abnormal way, and will not accept your solution even if your output is correct.

C++ users, please "return 0" manually in your main function. Python and Java users, as long as you do not call some system function to explicitly return a non-zero value, you should be fine.

2.2.3 Starter Code for C++, Java and Python

The following programs read an integer from the input, multiply it by 2 and print it to the output. This should teach you what a minimal solution should look like and how to do basic IO.

```
// C++
int main() // do not use void main!!!!
{
    int x;
    cin >> x;
    cout << x*2 << endl;
    return 0;
}
```

```
// Java
```

```
// do not specify any packages!!!
```

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

```
public class Main { // "Main" should be used for most online judges
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        int x = sc.nextInt();
        System.out.printf("%d\n", x);
    }
}
```

```
# python
x = int(input())
print(x*2)
```

2.3 Submitting Your Solution

See the Virtual Judge section.

3 Understanding The Judge's Verdicts

After your submission is received, the judge will run your program against it's own comprehensive test cases, which are not visible to you, and give you a verdict.

The judge's verdict can be one of the following types:

- **Accepted (AC):** Your solution has passed all test cases. Congrats!
- **Wrong Answer (WA):** Your solution is incorrect. The output does not match the judge's.
- **Time Limit Exceeded (TLE):** Your solution has exceeded the time limit.
- **Memory Limit Exceeded (MLE):** Your solution has used more memory than allowed.
- **Runtime Error (RE):** Your program crashed for whatever reason. (e.g. invalid memory access caused by array index out of bound)
- **Compile Error (CE):** Your code failed to compile.
- **Presentation Error:** This means your solution is correct but the output format is wrong. Some judges are strict about the output format. For example, they may not allow redundant spaces or redundant empty lines.

4 Working With Command Line Prompt

If you are using an EWS machine, your preferred IDE or GUI editor may not be available. In this case, you need to switch to a text editor (vim, emacs, sublime etc.), compile, run and test your code using the command line prompt.

4.1 Compile and Run Your Code

4.1.1 C++

```
# suppose src.cpp is your source file, use g++ to compile it
g++ src.cpp
# this will generate an executable file a.out, and you can run it by calling
./a.out

# you can use the -o option to specify the name of the executable
# otherwise, the name defaults to "a.out"
g++ src.cpp -o my_program

# if you want to use c++11, make sure to add the following option
g++ -std=c++11 src.cpp
```

4.1.2 Java

```
# suppose src.java is your source file, use javac to compile it
javac src.java
# this will generate a file named "src.class" under the same directory
# then use java to run the class file
java src.class
```

4.1.3 Python

```
# python is an interpreted language
# use python/python3 to run your code directly
python src.cpp
python3 src.cpp
```

4.2 Input/Output Redirection

Bash allows you to redirect the input and the output of a process. This is very useful for testing as it can save you from typing the input data every time.

```
# suppose you write your input data to a text file input.txt
# then you can use < to read the data from the file and feed it to a program
my_program < input.txt
```

```
# use > to write the output of a program to a file
my_program > output.txt
```

```
# use | to pipe the output of a program to another program
ls | grep '.cpp'
```

4.3 Other Useful Commands

```
# change the current directory to xxx
cd xxx
# the path can contain .. which means the "parent directory"
cd ../yyy

# display all the files and sub directories under the current directory
ls

# delete the file a.cpp
rm a.cpp
# delete the directory "mydir" recursively
rm -rf mydir
```