CSCI2100C Lab 1

Prepared by:

Lo Chun Hei

(office @ ERB615)

Welcome to CSCI2100C!

- Tutorial slides are uploaded to Blackboard 1 day before each tutorial.
- Remember to enroll in Piazza for discussion and questions!
- 4 individual programming labs (4% each)

Agenda

- Online Judge Platform
- Recommended IDEs
- C language
- Overview of Lab 1 Problems

Agenda

- **Online Judge Platform**
- Recommended IDEs
- C language
- Overview of Lab 1 Problems

Online Judge Platform (OJ)

Introduction

- Website: https://oj.boleyn.su/
- All 4 labs will be delivered via the OJ, i.e. the OJ handles
  - Posting of problems and their description
  - Submission of your codes
  - Judgement of correctness
    - You will get **full mark** on the problem only if your code passes **all** testcases (accepted);
    - **0** marks are given otherwise.
- Extra practice questions available apart from those in the 4 labs

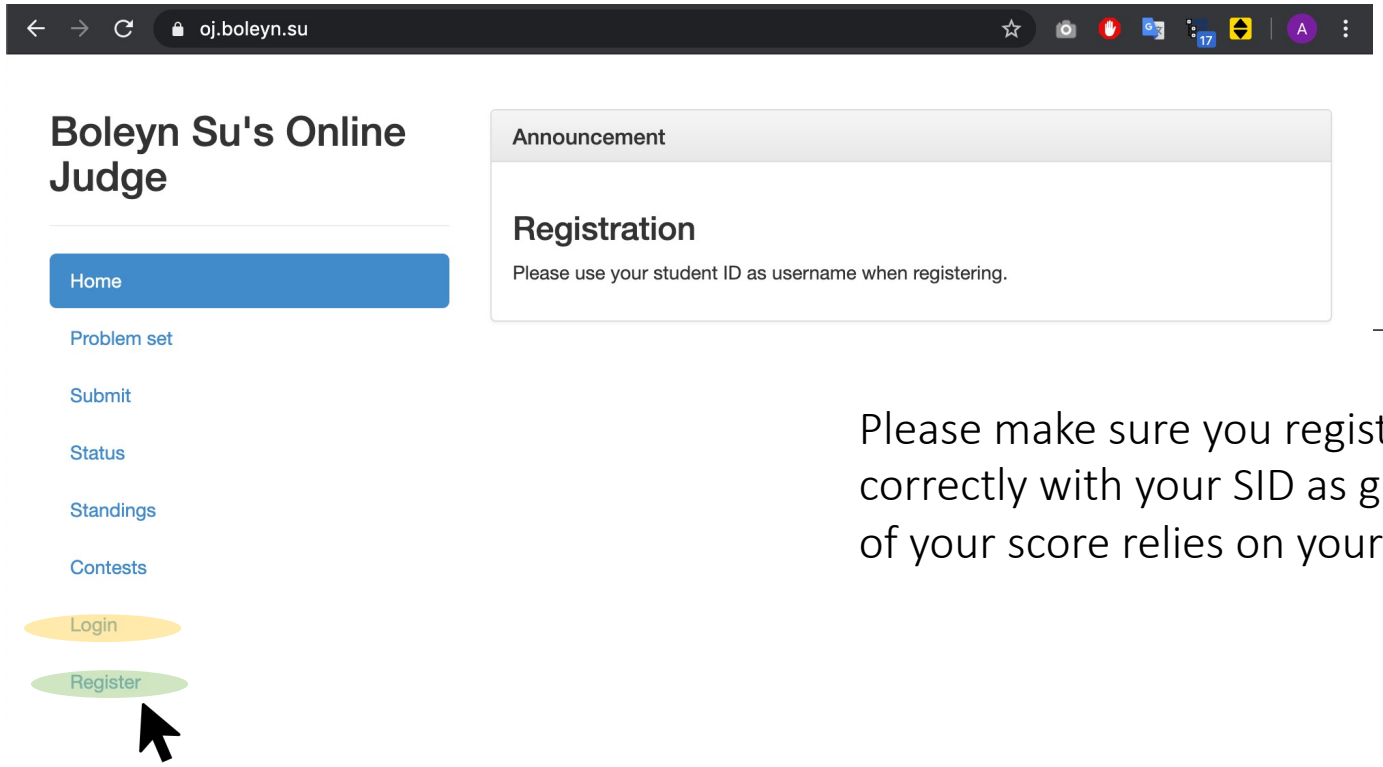
- If you have your laptop now, you may follow this hands-on guide on the OJ as we walk through it

Online Judge Platform (OJ)

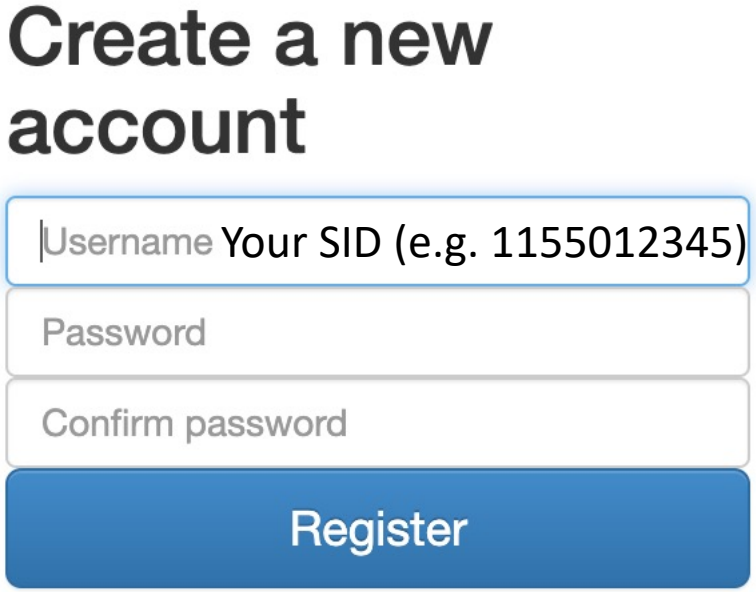
Guide on Usage

1. Registration/Login

- Click *Register* → Register → Click *Login* → Login



Please make sure you register correctly with your SID as grading of your score relies on your sid.

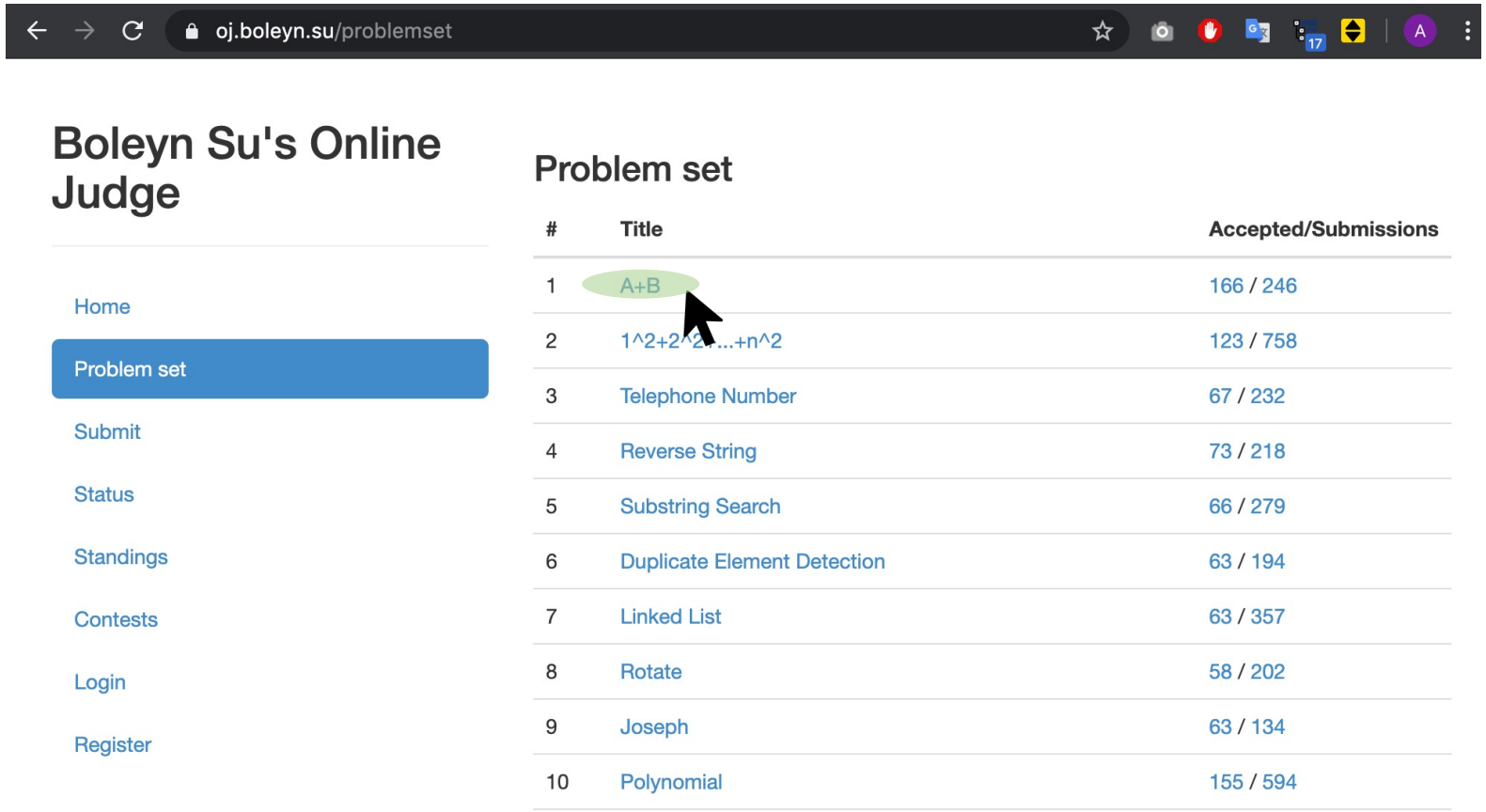


Online Judge Platform (OJ)

Guide on Usage

2. Problem Set

- List of problems you can try to solve!



- Click *Problem set*
- Click *A+B*

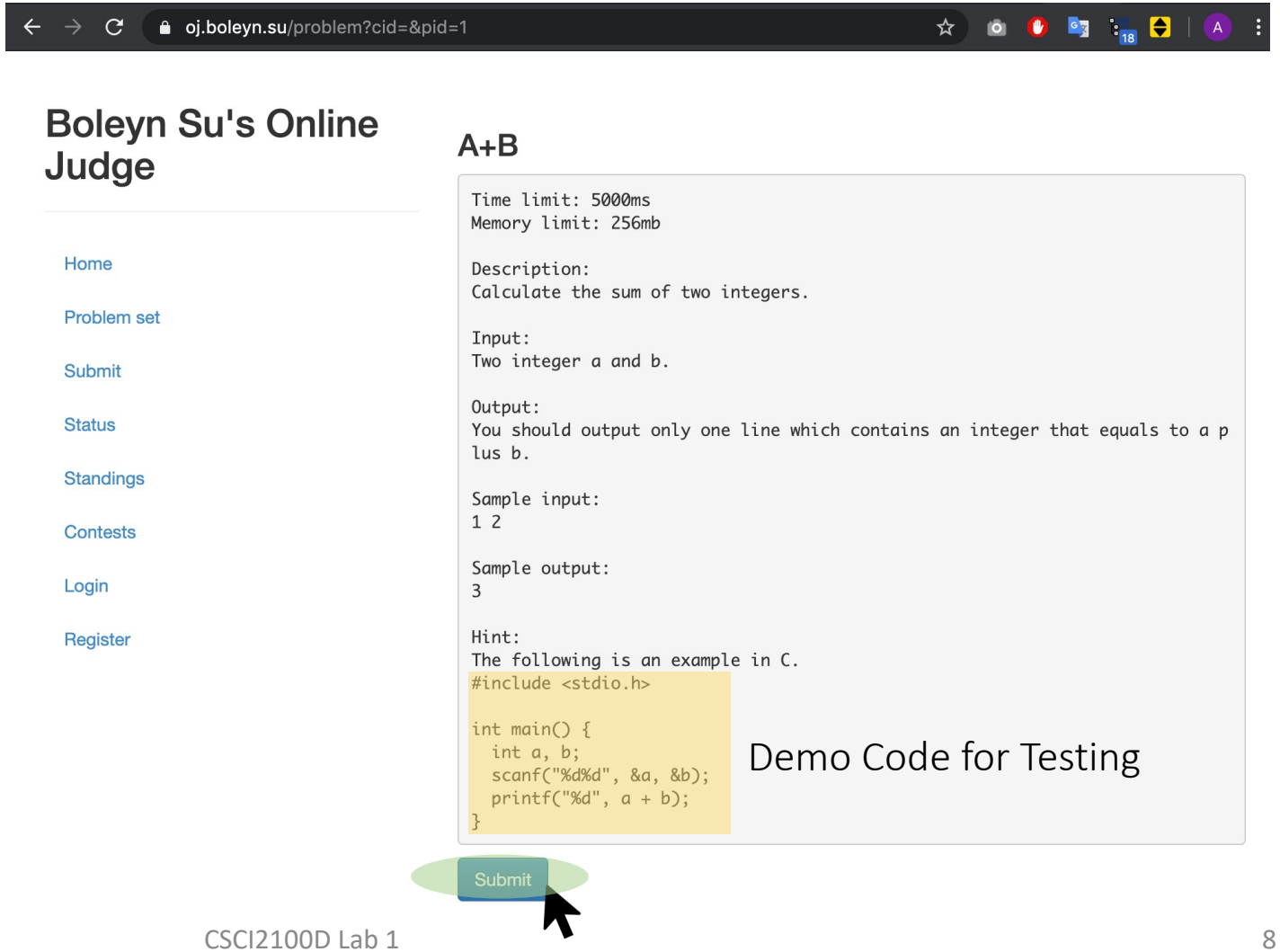
Online Judge Platform (OJ)

Guide on Usage

3. Problem Description

- Where the description, input/output specifications (with samples) of the problem are provided

- Copy the demo code
- Click *Submit*

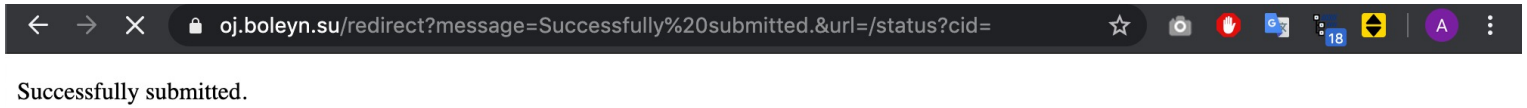


Online Judge Platform (OJ)

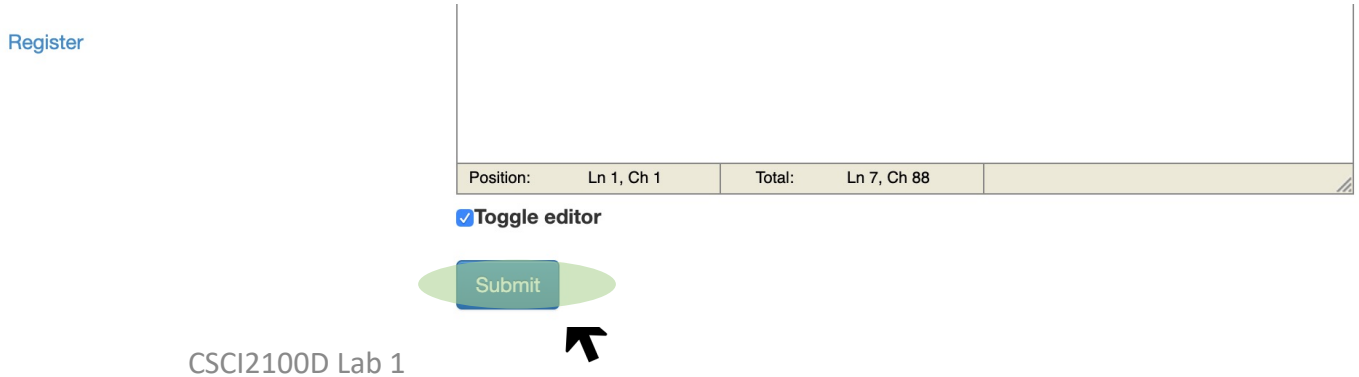
Guide on Usage

4. Code Submission

- Where you submit your code to the problem



- Paste the demo code
- Click *Submit*

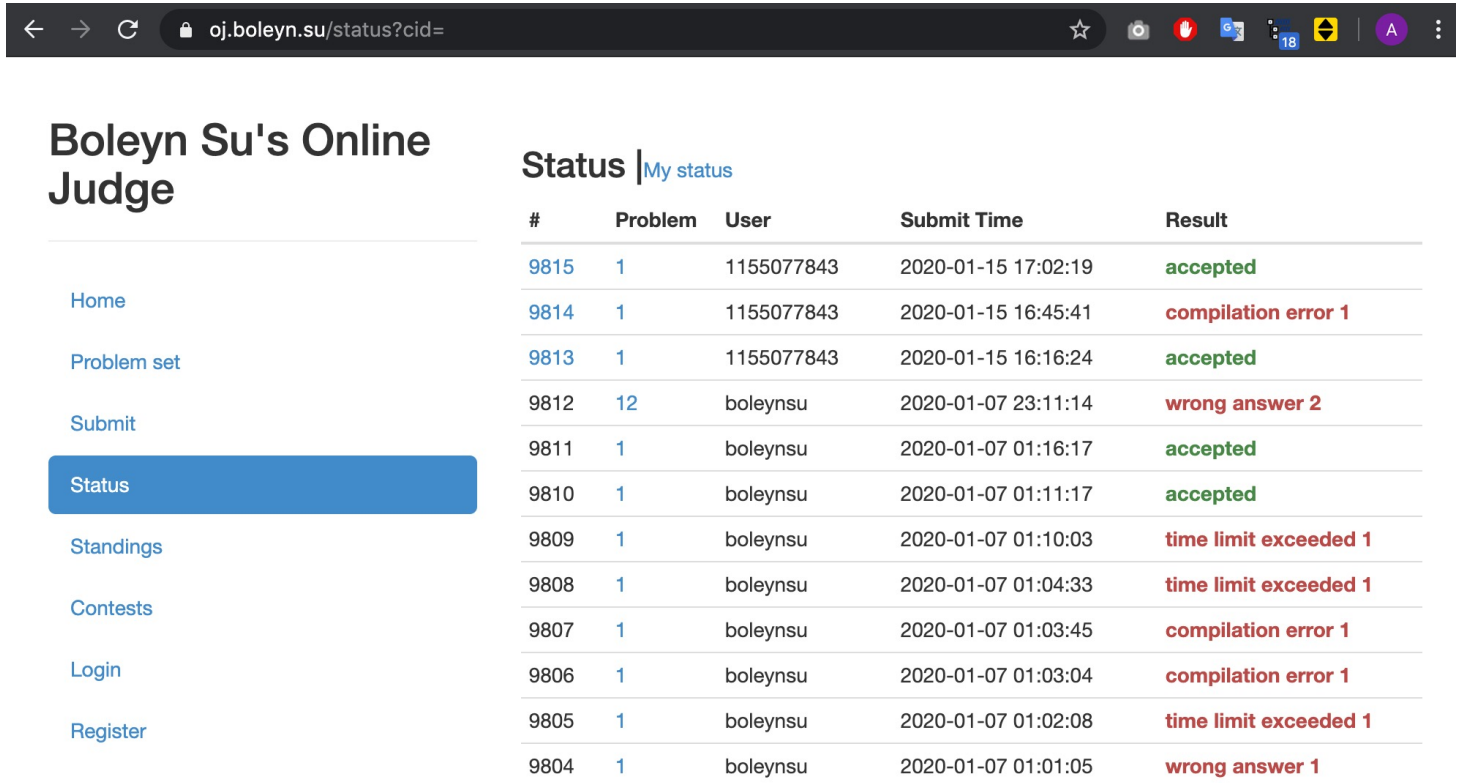


Online Judge Platform (OJ)

Guide on Usage

5. Submission Status

- Accepted:
  - It is all good: compiled successfully with correct answers on all testcases and executed within time limit
- Wrong answer #n:
  - Outputs of the n-th testcase do not match the correct output
- Time limit exceeded #n :
  - Your code runs too long for the n-th testcase. Normally the time limit is 5 seconds;
- Compilation error #n :
  - Your code does not compile (e.g. syntax error)
- Testing stops for the remaining testcases after the n-th one.



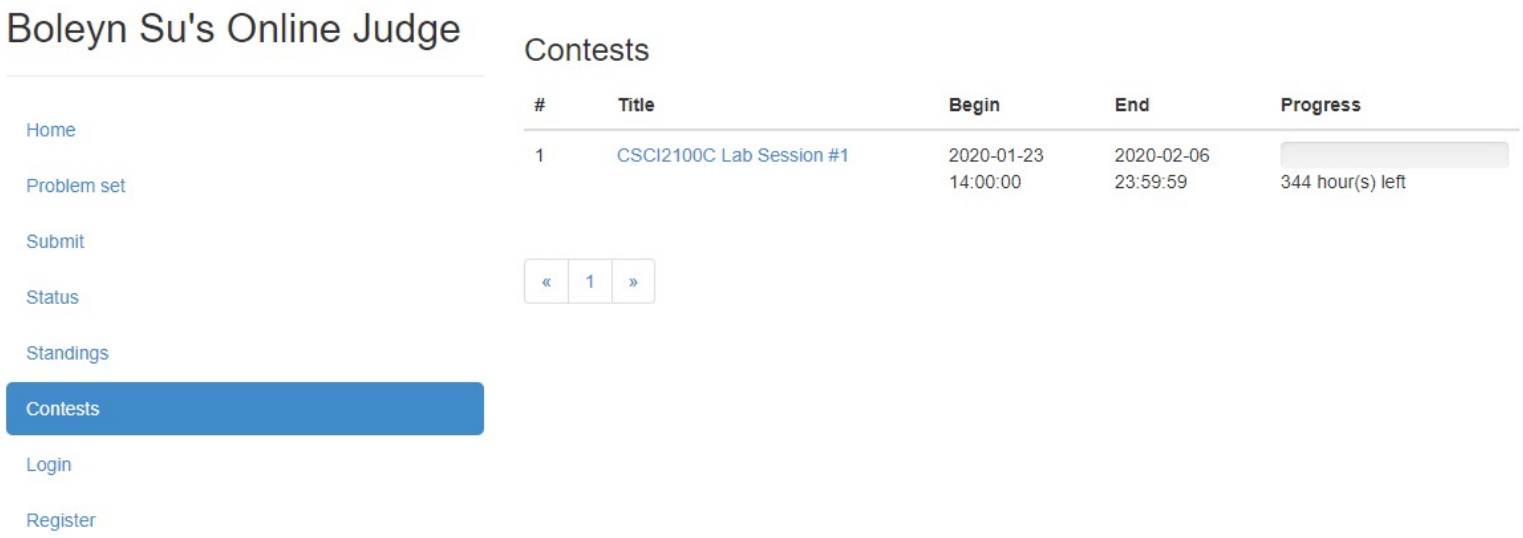
#	Problem	User	Submit Time	Result
9815	1	1155077843	2020-01-15 17:02:19	accepted
9814	1	1155077843	2020-01-15 16:45:41	compilation error 1
9813	1	1155077843	2020-01-15 16:16:24	accepted
9812	12	boleynsu	2020-01-07 23:11:14	wrong answer 2
9811	1	boleynsu	2020-01-07 01:16:17	accepted
9810	1	boleynsu	2020-01-07 01:11:17	accepted
9809	1	boleynsu	2020-01-07 01:10:03	time limit exceeded 1
9808	1	boleynsu	2020-01-07 01:04:33	time limit exceeded 1
9807	1	boleynsu	2020-01-07 01:03:45	compilation error 1
9806	1	boleynsu	2020-01-07 01:03:04	compilation error 1
9805	1	boleynsu	2020-01-07 01:02:08	time limit exceeded 1
9804	1	boleynsu	2020-01-07 01:01:05	wrong answer 1

Online Judge Platform (OJ)

Guide on Usage

6. Lab Assignments

- Each lab assignment will appear at *Contests* upon release of it
- After the deadline specified by *End*, no more submission will be accepted
- Click into the title to submit your codes



Agenda

- Online Judge Platform
- **Recommended IDEs**
- C language
- Overview of Lab 1 Problems

Recommended IDE

Recommended:

- Visual Studio Code (Windows/Mac)
  - https://code.visualstudio.com/
- CodeBlock (Windows/Mac)
  - http://www.codeblocks.org/downloads/26
- Dev C++ (Windows)
  - https://sourceforge.net/projects/orwelldevcpp/

Also Consider:

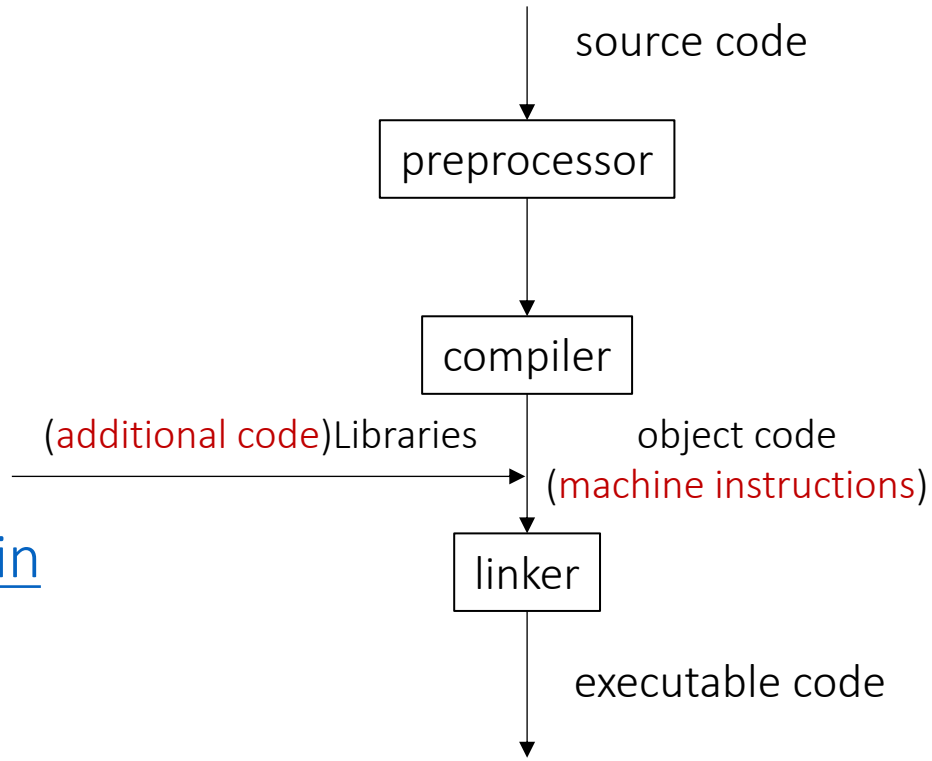
- Online C IDE
- Text editor + C compiler
- IDE of your choice

C language

Compilation Model

- Most frequently used and free available compiler:
  - GNU C compiler
- Installation:
  - Refer to https://www.tutorialspoint.com/cprogramming/c_environment_setup.htm
- Compile in terminal under the directory with your C source code, e.g.

```
$ gcc hello.c –o hello_program
$ ./hello_program
Hello, World!
```



Agenda

- Online Judge Platform
- Recommended IDEs
- **C language**
- Overview of Lab 1 Problems

C language

Revisit

- All programming labs must be completed with C only (not C++).
- Assume basic knowledge of C
  - Syntax, data type, function, variable scoping, control flow, loop, pointer, array, struct, header file, …
- Nice C tutorials can be found at:
  - https://www.learn-c.org/
  - https://www.tutorialspoint.com/cprogramming/index.htm
  - Google!
- Revisit some of the basics here

C language

Pointer

pointer_demo.c

```
#include <stdio.h>
int main () {
    int var = 20; /* actual variable declaration */
    int *ip; /* pointer variable declaration */
    ip = &var; /* store address of var in pointer variable*/
    printf("Address of var variable: %x\n", &var ); /* address stored in pointer variable */
    printf("Address stored in ip variable: %x\n", ip ); /* access the value using the
pointer */
    printf("Value of *ip variable: %d\n", *ip ); return 0;
}
```

Output

```
Address of var variable: bffd8b3c
Address stored in ip variable: bffd8b3c
Value of *ip variable: 20
```

C language

Struct(ure)

struct_demo.c

```c
#include <stdio.h>
#include <string.h>
struct Books {
    char title[50];
    int book_id;
};
void printBookTitle( struct Books *book ) {
    printf( "Book title : %s\n", book->title);
}
int main( ) {
    struct Books Book1; /* Declare Book1 of type Book */
    strcpy( Book1.title, "C Programming");
    printBookTitle( &Book1 ); /* print Book1 info by passing address of Book1 */
    return 0;
}
```

Output

```
Book title : C Programming
```

C language

Header File

- The C **preprocessor** scans the specified file as input before continuing with the rest of the current source file.
- An example is shown on the right, where a main program called *header_demo.c* uses the header file *header.h*

For system header files

```
#include <file>
```

For self-written header files

```
#include "file"
```

header.h

```
char *test (void);
```

header_demo.c

```
int x;
#include "header.h"

int main (void) {
    puts (test ());
}
```

Output of C preprocessor while compiling *header_demo.c*

```
int x;
char *test (void);

int main (void) {
    puts (test ());
}
```

C language

Abstract Data Type (ADT)

- Extending from the primitive data type in C, e.g. int, char, float, …
- Idea of ADT:
  - Implementation details are hidden
  - Able to use the ADT:
    - With the interface (commented usage)
    - Without knowing about the low-level logistics
- An **ADT** may be implemented in different **data structures**
  - E.g. Stack may be implemented with linked list or array

set.h

```
#include <stdio.h>
#include <stdlib.h>
struct Node {
    int Element;
    struct Node * Left;
    struct Node * Right;
}
struct Node * init_set(int x);
struct Node * set_insert(struct Node * r, int x);
void set_delete(Struct Node * r, int x);
```

set.c

```
#include "set.h"
struct Node * init_set(int x) {
    /* Implementation here */
}
struct Node * set_insert(struct Node * r, int x) {
 /* Implementation here */
}
void set_delete(Struct Node * r, int x) {
 /* Implementation here */
}
```

main.c

```
#include <stdio.h>
#include "set.h"
int main (void) {
    Node * u;
    ...
```

Agenda

- Online Judge Platform
- Recommended IDEs
- C language
- **Overview of Lab 1 Problems**

Lab 1

Problem 1

- Straight forward binary search
- Key to note:
  - Uniqueness of *n* integers
  - Condition for -1
  - Use the provided code template
    - Complete the missing part and submit

Binary Search

Description:

Given a list L of unique integers and a list T of target values, find out the indices of the target values in the list L (if exists), or report its absence.

Input:

First line contains a non-negative integer N;

The second line contains L, a sorted (in ascending order) list of N unique integers, L_0, L_1, ..., L_(N-1);

The third line contains T, a list of M target values, T_0, T_1, ..., T_(M-1), each separated by a space.

0 <= N,M <= 10^6

Output:

A line of M integers, I_0, I_1, ..., I_(M-1), where I_j = x if L_x = T_j, otherwise I_j = -1 if T_j is not in L.

Sample Input 1:

8

1 12 25 30 36 40 45 58

12 25 30 36 40 45 58 1 12

Sample Output 1:

1 2 3 4 5 6 7 8 0 1

Sample Input 2:

10

1 2 3 4 5 6 7 8 9 10

1 0 10

Sample Output 2:

0 -1 9

Lab 1

Problem 2

- Familiarize with the use of ADT
  - Just apply the described function(s) and ignore the underlying implementation
- Key to note:
  - Input may not necessarily be sorted
  - Choose the appropriate function(s) to apply
    - Description of function usage on OJ!
  - Use the provided code template
    - Complete the missing part and submit

The implemented ADT interface:

```
struct Node {
    int Element;
    struct Node * Left;
    struct Node * Right;
}
struct Node * init_set(int x);
struct Node * set_insert(struct Node * r, int x);
void set_delete(struct Node * r, int x);
struct Node * preorder_merge(struct Node * r, struct Node * r_new);
struct Node * set_union(struct Node * a, struct Node * b);
void print_set(struct Node * r);
void arr_union(int a[], int b[], int size_a, int size_b);
```

Set Union

Description:

Given two arrays of non-negative integers, denoted as arrA and arrB, the sets setA and setB are the unique elements of arrA and arrB respectively. Please complete the missing part of the following code to compute the union of setA and setB with the aid of the implemented ADT:

Input:

Four lines, where:

First line is a non-negative integer A;

Second line is arrA of A non-negative integers;

Third line is a non-negative integer B;

Fourth line is arrB of B non-negative integers;

0 <= A, B <= 10^3

Output:

You should output the union of setA and setB in ascending order.

Sample Input 1:	Sample Output 1:
7	1 2 3 4 5 6 7 8 9 10
1 2 5 6 7 9 10	
5	
1 2 3 4 8	

As an aside: What if *arrA* and *arrB* are both sorted? Can you think of an efficient algorithm to return *setA* ∪ *setB*?

Last but not Least

- Please add this declaration on top of (commented as shown) all your codes submitted to the OJ.

```
/*
I, <Your Full Name>, am submitting the assignment for
an individual project.
I declare that the assignment here submitted is original except for
source material explicitly acknowledged, the piece of work, or a
part
of the piece of work has not been submitted for more than one
purpose
(i.e. to satisfy the requirements in two different courses) without
declaration. I also acknowledge that I am aware of University
policy
and regulations on honesty in academic work, and of the
disciplinary
guidelines and procedures applicable to breaches of such policy and
regulations, as contained in the University website
http://www.cuhk.edu.hk/policy/academichonesty/.
It is also understood that assignments without a properly signed
declaration by the student concerned will not be graded by the
teacher(s).
*/
```

Question?