Apéndice A

Juez Virtual de la Carrera de Informática

A.1. Introducción

El juez virtual de la carrera de informática es un programa cuya finalidad es la de verificar programas desarrollados en Java y C,C++, sean correctos. Esta verificación se realiza en base de datos de prueba. El documento está destinado a la presonas quue quieren aprender a programar, para que puedan hacer uno uso efectivo del juez.

A.2. Comenzar

Para empezar debe dirigirse a la pagina del juez cuya dirección (figura:A.1) es: www.jv.umsa.bo.

En la página se visualizan pestañas: *Principal, problemas, estado, ranklist, concursos, FAQs* y en una posición izquierda superior derecha *Entrar Registro.*

A.3. Registrarse

En la parte superior derecha se cuenta con un enlace para realizar el registro (figura:A.1):

En la página de registro (figura: A.2 debe llenar todos los datos requeridos. Debe tener cuidado de proporcionar datos que recuerde, el nombre de usuario será con el que será identificado y la contraseña la clave para su ingreso. Es importante que proporcione

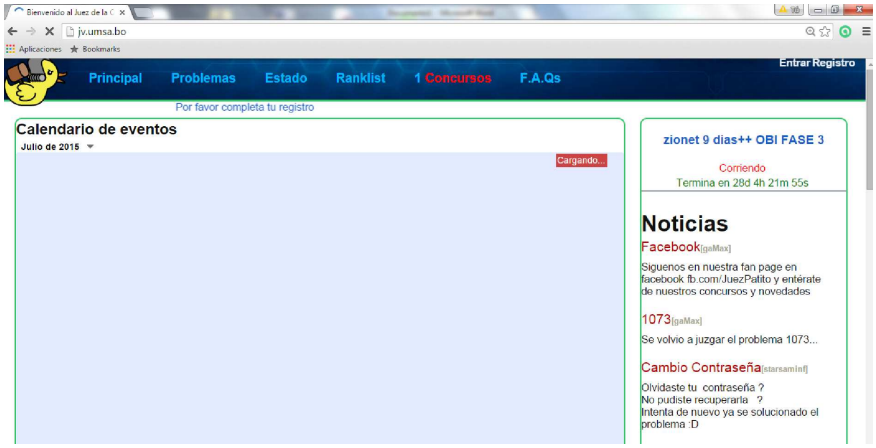

Figura A.1: Juez Virtual

un correo activo y no olvidar el captcha, para finalizar oprima crear y listo puede ingresar al juez y resolver problemas.

A.4. Ingresar con mi cuenta de usuario

Dirigirse la página principal y oprimir el enlace de *entrar* (figura: A.4)

- Ingrese su usuario o Nick con el que se registro en el juez.
- Ingresar su contraseña.
- presionar *Submit* para ingresar.

Una vez que haya ingresado al juez notara que su usuario aparecerá en la esquina superior derecha

A.5. Envío de problemas

Muy independiente de los concursos se tienen la opción de resolver todo el conjunto de problemas disponibles en el juez. Para lo cual se debe dirigir a la parte de problemas:

Para poder enviar un problema (figura A.3) se debe ingresar al enunciado del problema por ejemplo resolvamos el *problema 1000 A+B*.

Cada problema tiene varios conceptos que describimos

- Descripción.- Describe el enunciado del problema.

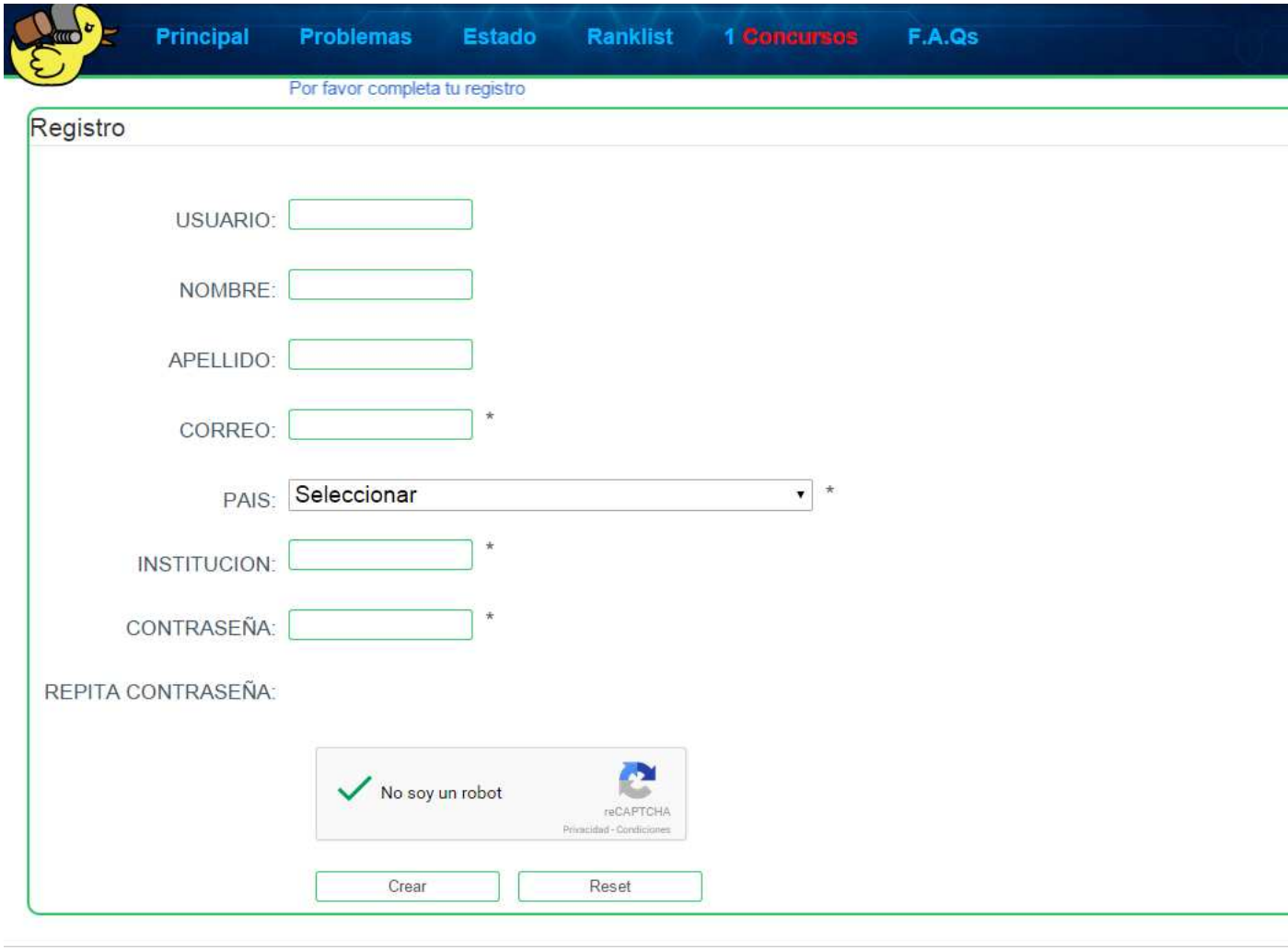

Figura A.2: Registro de usuario nuevo

- Entrada.- Describe la forma en la que están los datos de entrada.
- Salida.- Describe como se espera que presente la solución del problema.
- Ejemplo de entrada.- Este es un ejemplo de como son sus datos de entrada.
- Ejemplo de salida.- Esta es la solución en el formato esperado para los datos de entrada.

En el problema 1000 que resolveremos nos muestra la información de la imagen A.4

Si leemos el enunciado nos indica que debemos leer dos números A, B y luego imprimir el resultado. El programa debe leer los datos del teclado y mostrar el resultado

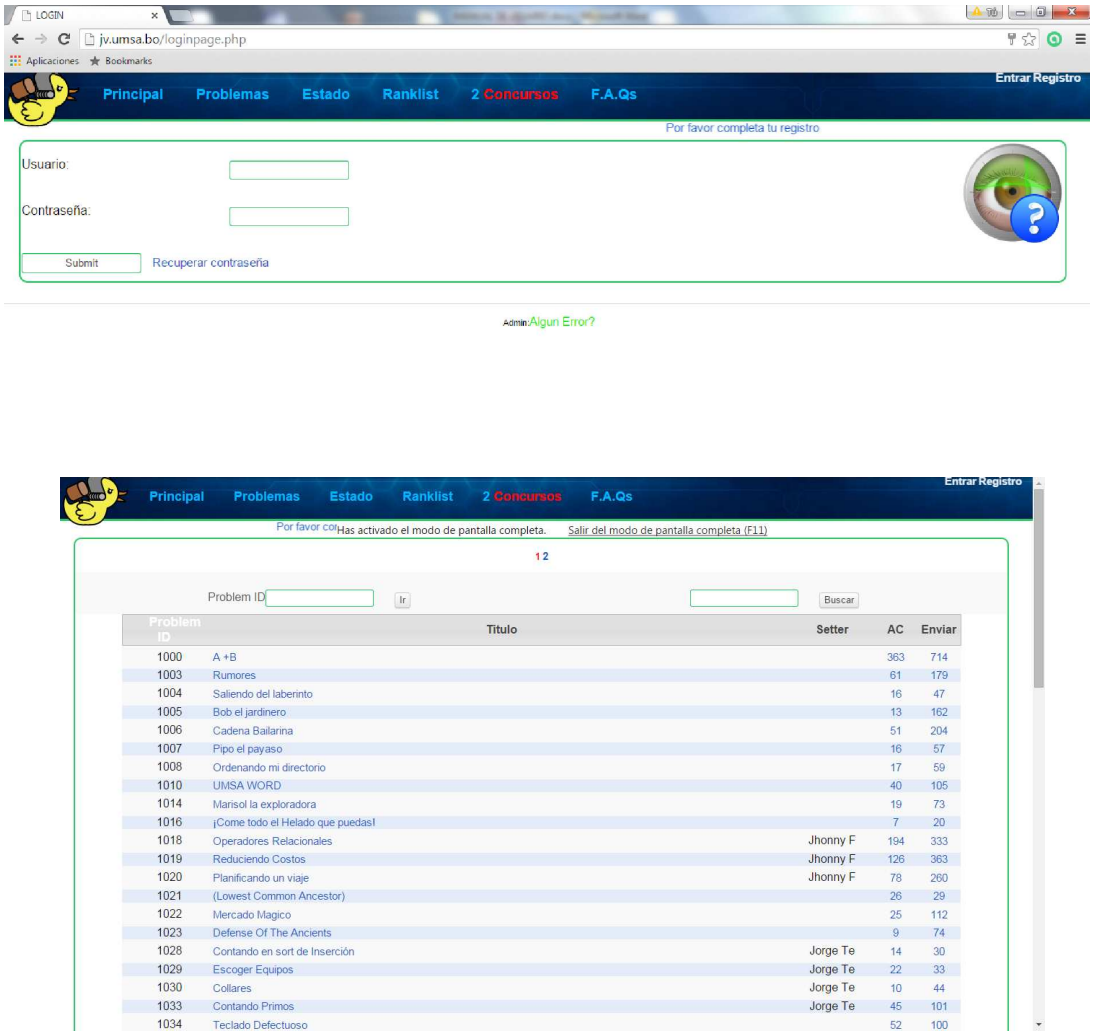

Figura A.3: Envío de problemas

en la pantalla. No se deben incluir mensajes tales como *ingrese A*, *la respuesta es*, etc. La respuesta debe tener un formato idéntico al presentado en el ejemplo.

Para enviar una solución (figura: A.5) primero se debe escoger el lenguaje de programación, que se encuentra al centro de la pantalla. En el caso de Java la clase debe tener el nombre de *Main* y el método principal debe llamarse *main*. Hay que recalcar que el programa debe escribirse en un solo archivo.

En segundo lugar copiamos el programa en recuadro y se debe oprimir el enlace de *ENVIAR*.

El programa que resuelve el problema 1000 es el siguiente:

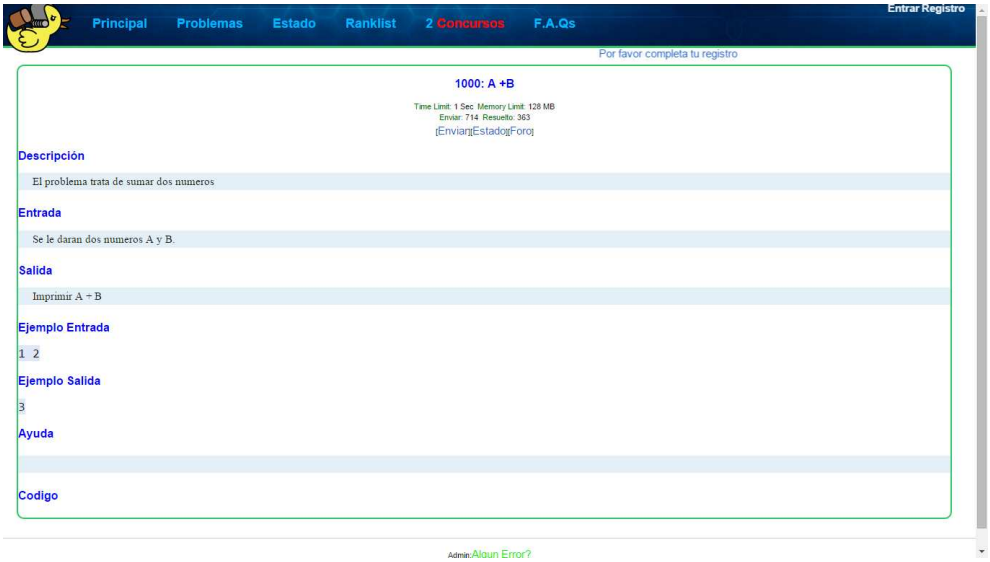

Figura A.4: Enunciado del ejercicio de prueba

```
1  import java.util.*;
2
3  public class Main{
4      public static void main(String args[]){
5          Scanner cin = new Scanner(System.in);
6          int a, b;
7          while (cin.hasNext()){
8              a = cin.nextInt(); b = cin.nextInt();
9              System.out.println(a + b);
10             }
11     }
12 }
```

La clase *Scanner* es la que nos permite leer los datos del teclado. El método *hasNext()* permite saber si existen más datos en el archivo de entrada. Esto asociado a la instrucción **while** nos permite procesar todos los datos de la entrada.

Una vez copiado el código y presionada la tecla *ENVIAR*, se evaluará la solución, las posibles respuestas que nos puede dar el juez son:

- Pending.- Indica que aún esta pendiente de revisión. Puede ser que el juez tenga muchos envios.
- Pending Rejudge.- Rejudge, significa que se ha decidió volver a evaluar los envíos de un problema especifico. Esto ocurre cuando los autores de los problemas des-

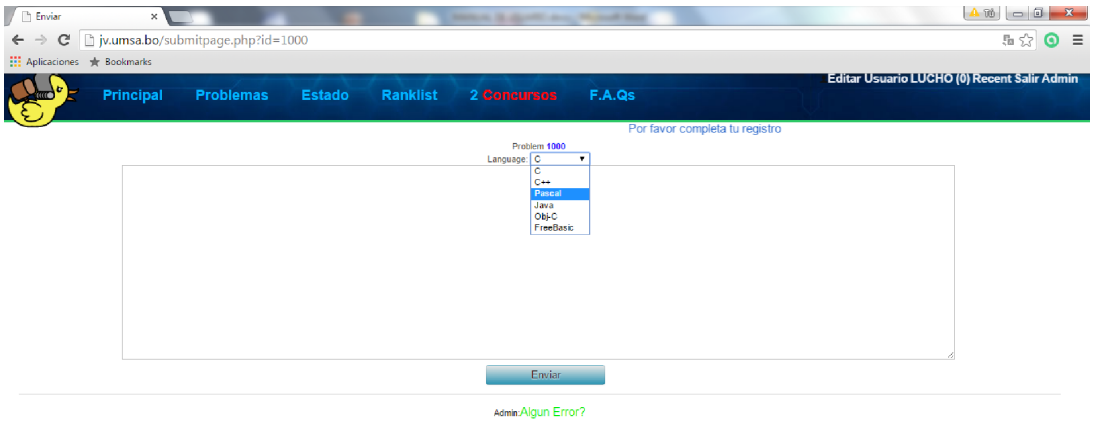

Figura A.5: Envió de un programa

cubren errores en los datos de prueba y por lo tanto es necesario, volver a revisar todos los envíos. Esto puede hacer que programas aceptados, luego se consideren fallidos y también problemas no aceptados ahora se consideren correctos.

- Compiling.- Indica que el programa esta siendo compilado.
- Running & Rejudging. Running indica que el programa esta siendo procesado y rejudging que esta volviendo a juzgar.
- Accepted.- Esto es lo que esperamos. La solución que enviamos es correcta y ha sido aceptada.
- Presentation Error.- Este error se presenta cuando hay espacios adicionales en la respuesta. Esto puede ser uno o más espacios al final de la linea. Una línea en blanco que falta o sobra en una respuesta, etc. Significa que eliminando los espacios probablemente se obtenga el resultado esperado.
- Wrong Answer. Indica que los resultados generados por el programa son diferentes a los registrados en el juez.
- Time Limit Exceeded.- Se da cuando el tiempo de proceso es mayor al esperado. Esto ocurre en los casos que existe una solución más eficiente que la desarrollada. Tambien si su programa esta en un ciclo y no termina.
- Memory Limit Exceeded.- Los programas generalmente deben resolverse utilizando 64kb o 128kb de memoria ram si exceden esto se produce el error.
- Output Limit Exceeded.- Cuando el archivo de salida es mayor que el permitido se obtiene el error.

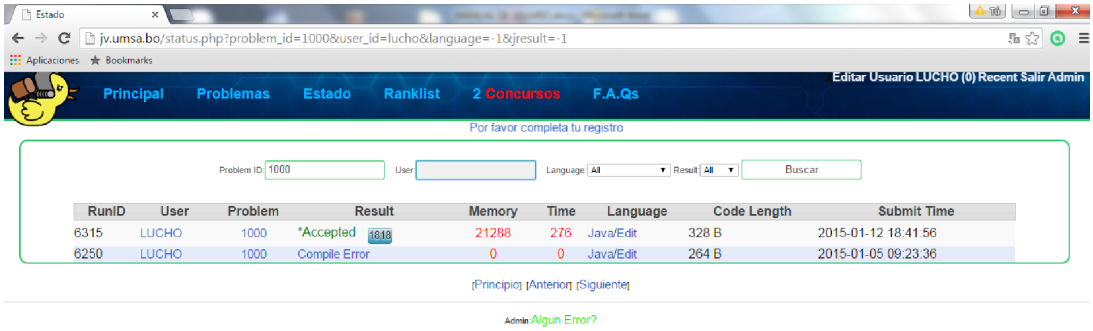

Figura A.6: Estado de un problema enviado

- Compile Error.- Significa error de compilación. En java puede indicar que su clase principal no es *Main*.

Para revisar el estado (figur A.6 de su envío acceda al menú donde indica *ESTADO* puede revisar su envío ingresando su usuario y el número del problema.

A.6. Competencias

Periódicamente, ya sea en clases o como tarea se crean competencias . Estas competencias son listas de problemas que se espera que deben resolver en un tiempo especifico. Esta opción tiene varias pestañas:

- Standings.- Esta opción nos nuestra es el detalle de problemas enviados, resueltos, y el numero de intentos para resolver un problema.
- Statistics.- Muestra los problemas indicando como fueron resueltos, cuales fueron resuelto, y otras estadistas.
- Enviar una solución.- Para esto solo ingrese el nombre del problema y tendrá todas las opciones para enviar problemas tal como se explico anteriormente.

Para establecer el orden de los participantes se toma la cantidad de problemas resueltos y el tiempo que demoró en resolver los problemas. En clases solo nos interesa el numero de problemas resueltos y no así el numero de intentos.

Apéndice B

Errores comunes en tiempo de ejecución

Durante la ejecución de un programa pueden producirse diversos errores. Se muestran los más comunes.

ArrayIndexOutOfBoundsException

Se produce cuando intentamos acceder a un vector (array) o cadena (String) con un index inválido (menor que 0 o mayor que su longitud .). Por ejemplo si definimos un vector de tamaño n el recorrido del mismo debe ser desde 0 hasta un $n-1$. Eso completa los n valores del tamaño del mismo. El recorrido correcto es:

```
for (int i= 0; i<n; i++){
.....
}
```

NumberFormatException

Se obtiene cuando un método convierte una cadena a un número y esta cadena no puede ser convertido. Cuando usamos *Integer.parseInt* y el argumento no es un numero entero. Lo mismo puede ocurrir al leer datos del flujo de entrada. El siguiente comando proporcionara éste error, dado que, el argumento contiene elementos que no son números.

```
Integer.parseInt("a123")
```

StackOverflowError

Ocurre típicamente cuando un método recursivo excede la capacidad de memoria para almacenar los valores intermedios.

NullPointerException

Ocurre cuando intentamos acceder a un objeto con una variable de referencia cuyo valor actual es null.

NoSuchElementException

Ocurre cuando tratamos de leer más allá del final de un archivo. Cuando leemos datos del teclado no ocurre porque siempre podemos seguir ingresando datos.

Índice de figuras