Olimpiada Informática de Extremadura

Escuela Politécnica – Cáceres
Universidad de Extremadura

MANUAL DE MOOSHAK

La prueba de programación de la **Olimpiada Informática de Extremadura** se realizará utilizando un juez automático llamado Mooshak. Esta es una aplicación que permite comprobar si un programa resuelve correctamente el problema planteado. Os recomendamos que preparéis la prueba "entrenando" con esta aplicación. De esa forma, el día del concurso, todo resultará más cómodo. A continuación, encontraréis unas sencillas indicaciones para utilizar Mooshak.

Índice de contenidos

1. Registro.

Para poder utilizar la aplicación es necesario, en primer lugar, registrarse como usuario. En la página principal de la Olimpiada (www.oiex.org) debéis elegir la opción "Preparación", allí encontraréis un enlace al "juez on_line". Inmediatamente aparecerá la siguiente página en la que debéis seleccionar la opción para registrarse (está señalada con una flecha roja en la figura 1).

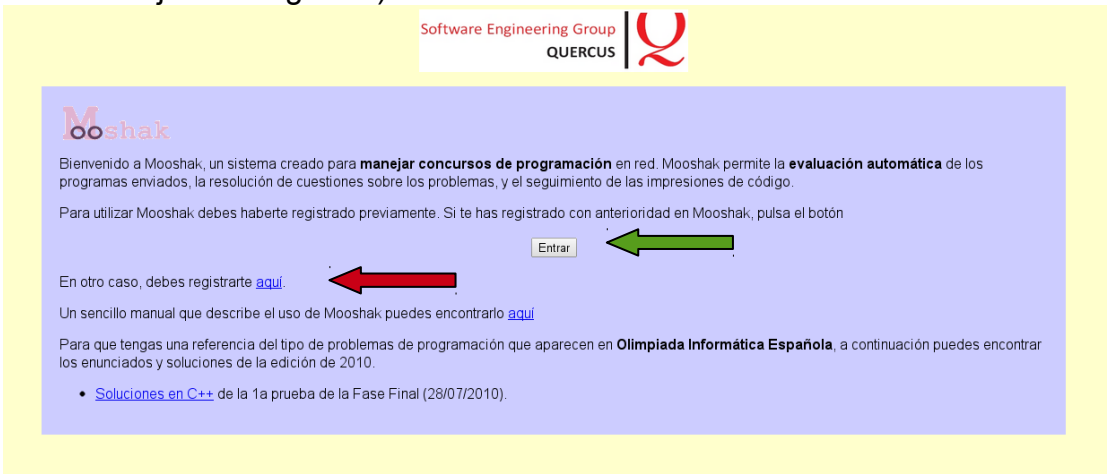


Figura 1. Mensaje de bienvenida

A continuación deberéis rellenar un sencillo formulario:

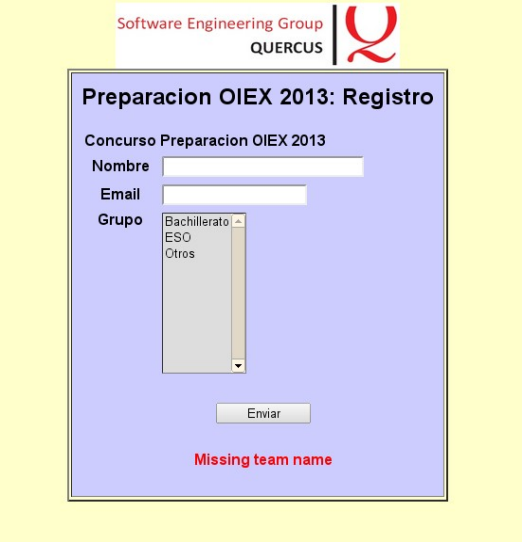


Figura 2. Formulario para el registro en la aplicación

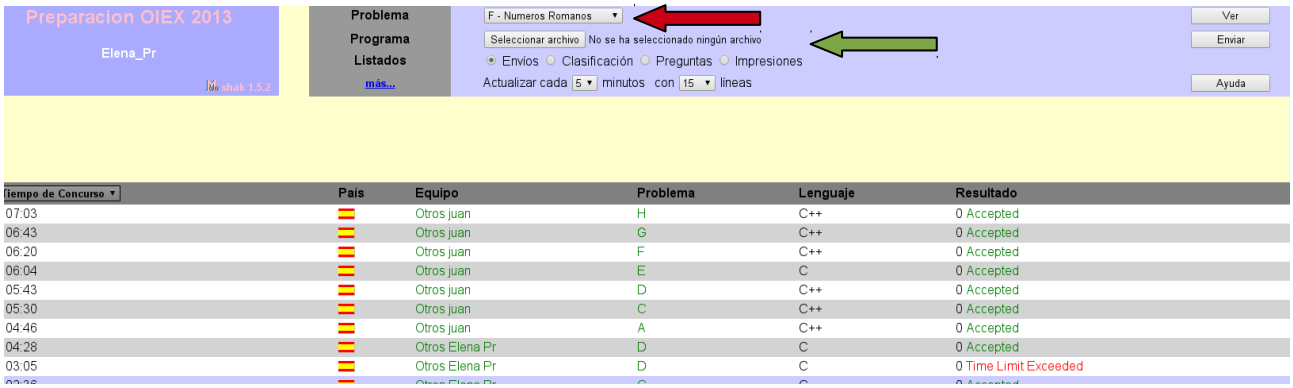
Importante: al utilizar este formulario, debéis introducir el nombre por el que "Mooshak" os va a conocer. Debéis tener en cuenta que Mooshak sólo aceptará letras, dígitos y el guión bajo "_", pero no acepta espacios en blanco, vocales con tilde, ñ's, o cualquier otro carácter "especial". Por lo tanto, tu nombre puede ser "PepePerez" o "Pepe_Perez" pero no "José Mª Pérez".
Inmediatamente, recibirás en tu correo electrónico un mensaje en el que se indicará la contraseña que necesitas para utilizar la aplicación.

2. Utilización de Mooshak.

Cuando tengas tu contraseña para entrar en la aplicación, deberás volver a la pantalla de entrada a "Mooshak" y pulsar el botón "entrar" (está señalada con una flecha verde en la figura 1). A continuación deberás rellenar un nuevo formulario con tu nombre de usuario y tu contraseña.

Figura 3. Formulario de entrada a la aplicación

Si eliges la opción de invitado puedes entrar en la aplicación sin haberte registrado previamente. Así podrás ver los resultados obtenidos por los diferentes participantes pero no podrás ver los enunciados de los problemas ni participar en el concurso.
Si has entrado como un usuario registrado, la pantalla que vas a encontrar es la siguiente:



Tiempo de Concurso	País	Equipo	Problema	Lenguaje	Resultado
07:03		Otros juan	H	C++	0 Accepted
06:43		Otros juan	G	C++	0 Accepted
06:20		Otros juan	F	C++	0 Accepted
06:04		Otros juan	E	C	0 Accepted
05:43		Otros juan	D	C++	0 Accepted
05:30		Otros juan	C	C++	0 Accepted
04:46		Otros juan	A	C++	0 Accepted
04:28		Otros Elena Pr	D	C	0 Accepted
03:05		Otros Elena Pr	D	C	0 Time Limit Exceeded

Figura 4. Pantalla de "trabajo" de Mooshak

En primer lugar deberás decidir cuál es el problema que quieres resolver, para ello podrás utilizar la pestaña señalada en la figura 4 con la flecha roja, en la parte inferior de la pantalla podrás ver el enunciado de dicho problema.
Deberás resolver el problema con un programa escrito en C, C++ o Java. En el apartado siguiente te daremos algunos consejos para realizar estos programas. Una vez escrito el programa deberás enviar el fichero que hayas creado a Mooshak. La pestaña "Seleccionar archivo" que señala la flecha verde te permitirá seleccionar tu fichero con comodidad dentro del sistema de carpetas de tu ordenador. Cuando hayas seleccionado el fichero, pulsa el botón "Enviar". Inmediatamente, verás el resultado en la parte inferior de la pantalla. Si el programa es correcto aparecerá el mensaje "Accepted" en color verde y si no lo es aparecerá un mensaje en color rojo.
Si quieres tener más información sobre estos mensajes de error, pulsa la tecla "Ayuda" y encontrarás las respuestas a las preguntas más frecuentes.
La opción "Ver" te permite ver el enunciado del problema seleccionado.
La opción "Preguntar" te permite enviar una pregunta al administrador de la aplicación.
La opción "Salir", evidentemente, te permite salir de la aplicación.

3. Algunos consejos para resolver los problemas correctamente

Para que Mooshak sepa qué lenguaje has utilizado para resolver el problema, debes darle al fichero que contiene la solución un nombre con la extensión adecuada.

Lenguaje	Extensión	Ejemplo de nombre para el fichero
C	cc	solucion.c
C++	cpp	solucion.cpp
Java	java	solucion.java

Para comprobar si un programa resuelve correctamente el problema propuesto, Mooshak lo ejecuta y compara la salida con una serie de casos de pruebas que están previamente definidos. Por este motivo, es muy importante que la entrada y la salida del programa se ajusten a lo indicado en el enunciado. Consejos:

- No dejes espacios en blanco o líneas en blanco que no estén indicadas en el enunciado.
- Cuida los caracteres utilizados en los mensajes. Por ejemplo, no es lo mismo escribir:
  - "5 es primo"
  - "5 es Primo"
  - "5 es primo"
- No es necesario utilizar ningún archivo de entrada o salida. La entrada se lee desde teclado, es decir, desde la entrada estándar. La salida se escribe en la pantalla (salida estándar). Ejemplos de instrucciones de entrada y salida:

Lenguaje	**instrucción de entrada**	**instrucción de salida**
C	`scanf("%d", &A);`	`printf ("%d\n", A*2);`
C++	`cin >> A;`	**`cout << A*2 << endl;`**
java	**usar `System.in.readLine();`**	**`System.out.println(A*2);`**

- No debéis utilizar mensajes que actúen como interfaces de usuario como por ejemplo "Introduzca el dato: ".
- No utilicéis funciones como **getch()** en C, ya que éstas esperan a que un usuario presione una tecla y el juez automático considerará que el programa está "parado".
- También es muy importante que utilicéis el menor número posible de librerías.

Cualquier infracción a estas reglas puede representar un error para el evaluador automático. Esto puede ocurrir aunque el programa funcione correctamente con tu compilador habitual. Por lo tanto, ten en cuenta estas recomendaciones al resolver los problemas.

4. Ejemplo de problema resuelto

A continuación presentamos la solución, en diferentes lenguajes de programación, de un problema muy sencillo.

Enunciado del problema
Escribir un programa que, dado un número entero, escriba el doble de dicho número.

Entrada:

La entrada consta de varios casos de prueba. La primera línea indica el número de casos. Las líneas restantes tienen un número entero cuyo doble hay que calcular.

Salida:
Por cada caso de prueba, el programa debe escribir el doble del número indicado.

Ejemplo de entrada:	Ejemplo de salida:
4	
9	18
16	32
-5	-10
3	6

En las soluciones que aparecen a continuación todos los programas tienen la misma estructura. Además, hemos utilizado, en todos los casos, las siguientes variables:

- **num_datos**, se utilizará para almacenar el valor de la primera línea, es decir, el número de casos de prueba
- **dato**, almacenará, de forma secuencial, el resto de los números que constituyen la entrada del programa, es decir, los números cuyo doble vamos a calcular
- **i,** es una variable *índice* que utilizaremos en el bucle encargado de leer y procesar los datos.

Solución en C:

```
#include <stdio.h>
int main(){
     int num_datos, dato, i;
     scanf(“%d”, &num_datos);
     for (i = 0; i < num_datos; i++){
          scanf(“%d”, &dato);
          printf(“%d\n”, dato * 2);
     }
     return 0;
}
```

Solución en C++:

```
#include <iostream>
using namespace std;
int main(){
     int num_datos, dato, i;
     cin >> num_datos;
     for (i = 0; i < num_datos; i++){
          cin >> dato;
          cout << dato * 2<< endl;

     }
     return 0;
}
```

Solución en Java:

```
import java.io.BufferedReader;
import java.io.InputStreamReader;

@SuppressWarnings("all")
public class MultiplicarPorDos{
 public static void main(String[] args) throws java.io.IOException{
  BufferedReader in;
  in = new BufferedReader(new InputStreamReader(System.in));
  int num_datos = 0;
  num_datos = Integer.parseInt(in.readLine());
  for( int i = 0; i < num_datos; i++ ){
     int dato = Integer.parseInt(in.readLine());
     System.out.println(dato*2);
  }
 }
}
```

5. Errores frecuentes

Cuando envías a Mooshak un programa que resuelve un determinado problema, tu objetivo será que aparezca el mensaje "Accepted". Eso significa que el programa ha funcionado según lo esperado. Te recomendamos que antes de enviar el programa hagas alguna prueba para comprobar que su funcionamiento es correcto ya que, en la competición, los envíos incorrectos serán penalizados.
Sin embargo, en ocasiones, recibirás un mensaje que te índica la existencia de un problema. Estos son algunos de los errores que pueden aparecer:

Compile Time Error
El sistema no ha podido compilar (traducir) el programa debido a algún error gramatical. Si haces "doble click" sobre el mensaje, podrás ver otros mensajes que explican la situación (indicando habitualmente la línea en la que se ha detectado el problema), y que te ayudarán a encontrar el error. Habríamos tenido un error de este tipo si hubiéramos enviado el siguiente código para el ejemplo del apartado 4. Un despiste al escribir el nombre de una variable hace que el compilador no la reconozca.

```
#include <iostream>
using namespace std;
int main(){
     int num_datos, dato;
     cin >> num_datosss;
     for (int i = 0; i < num_datos; i++){
          cin >> dato;
          cout <<dato * 2<<endl;
     }
     return 0;
}
```

El mensaje de ayuda que aparecería sería el siguiente:
error: 'num_datosss' was not declared in this scope

Este tipo de errores se pueden evitar, compilando previamente el programa antes de enviarlo.

Runtime Error
El programa ha sido compilado con éxito pero la ejecución ha fallado.
El siguiente ejemplo provocaría un error de este tipo. El programa es gramaticalmente correcto y se compila sin problemas. Sin embargo, la ejecución de la línea destacada en rojo (concretamente: `dato/i`) provocaría un error, ya que el primer valor que toma la variable **i** es 0 y no es posible dividir por ese número.

```
int main(){
     int num_datos, dato;
     cin >> num_datos;
     for (int i = 0; i < num_datos; i++){
          cin >> dato;
          cout << dato / i <<endl;
     };
     return 0;
}
```

Time Limit Exceeded
El programa ha tardado en ejecutarse un tiempo mayor que el disponible. Revisa el código, puede ser muy poco eficiente o puede que tengas un bucle mal definido y que no acaba nunca.

Wrong Answer
El programa se ha compilado y ejecutado correctamente, pero la salida que ha generado no coincide con la esperada por el juez. Por ejemplo, el programa que aparece a continuación funciona perfectamente y, sin embargo, no resuelve el problema planteado ya que no multiplica los datos por 2 sino por 3. En este caso es posible que no se haya entendido bien el enunciado del problema.

```
int main(){
     int num_datos, dato;
     cin >> num_datos;
     for (int i = 0; i < num_datos; i++){
          cin >> dato;
          cout <<dato * 3<<endl;
     };
     return 0;
}
```

Presentation Error
La salida del programa está muy cerca de la salida esperada, pero hay algún pequeño detalle que las diferencia. Estás muy cerca de la solución, probablemente has escrito algún espacio o alguna línea en blanco que Mooshak no espera. En el siguiente ejemplo,

el problema está en la línea destacada en rojo que escribe en la pantalla un salto de línea inesperado.

```
int main(){
     int num_datos, dato;
     cin >> num_datos;
     for (int i = 0; i < num_datos; i++){
          cin >> dato;
          cout <<dato * 2<<endl;
     }
     cout << endl;
     return 0;
}
```

Invalid Submission
El envío no ha sido correcto. Es posible que el fichero enviado no tenga un nombre válido.

Output Limit Exceeded
La salida que genera el programa es demasiado grande.

Program Size Exceeded
El código del programa es demasiado largo.

Invalid function
Se ha violado alguna regla de Mooshak, por ejemplo, utilizar librerías no estándar o terminar el programa con “return 1” o “exit 1”.