
Práctica de PRO2 - Primavera 2023 - entrega final definitivaX77099_es

Este problema es el único canal disponible para realizar **la entrega definitiva** de la práctica y también **la entrega de los materiales para evaluar la competencia transversal “Treball en equip”**. Tened en cuenta que

- El plazo de entrega acaba el martes 23 de mayo, a las 22h.
- Esta es la entrega que dará lugar a la mayor parte de la nota de la práctica (90%); la nota de la entrega tendrá una parte procedente de corrección automática y otra procedente de corrección manual (consultad los detalles en el pdf sobre evaluación en el apartado “Pràctica” de la web de PRO2)
- No superar ningún juego de pruebas conllevará un cero en la nota de la práctica
- En esta entrega pedimos más cosas, que detallamos a continuación **en negrita**, además del código de la práctica y el Makefile, de las que se pedían en la entrega provisional
- Pueden realizarse tantas entregas como se quiera, pero solo se tendrá en cuenta la más reciente que no sea SE (“Setter Error”).
- El problema de la entrega final provisional seguirá disponible hasta horas antes del fin del plazo de entrega. De hecho, recomendamos usarlo para probar el código mientras no tengáis preparado el material adicional que pedimos

Observación

El Jutge prueba vuestras entregas mediante 4 juegos de pruebas:

- sample: el juego de pruebas público
- privat1: combinación de todos los juegos de pruebas de la entrega intermedia
- privat2: un poco de todo, salvo eficiencia
- privat3: eficiencia

En un fichero llamado `practica.tar` tenéis que entregar

- Los ficheros `.hh` y `.cc` de las clases y el programa principal **con los comentarios Doxygen correspondientes**. No incluyáis ficheros `.o`, `.exe` y similares
- El fichero **Makefile**, que usaremos para generar y probar el ejecutable; un Makefile incorrecto puede dar lugar a malos resultados en el Jutge o a penalización en la nota manual
- Un fichero `html.zip`, **obtenido comprimiendo la carpeta html del Doxygen generado a partir de los `.hh` y `.cc` anteriores**. Dicha documentación no solo ha de incluir la parte pública de las clases (como en la entrega de la especificación) sino también la parte privada (atributos y métodos privados) y la implementación de todas las operaciones

- Un único integrante de cada equipo ha de entregar un fichero .zip con los materiales requeridos para evaluar la competencia “Treball en equip”, siguiendo las instrucciones contenidas en el documento correspondiente, disponible en el apartado “Pràctica” de la página web de la asignatura

Tened en cuenta las siguientes restricciones:

- El fichero que contiene el programa principal se ha de llamar `program.cc`
- El Makefile ha de generar un ejecutable llamado `program.exe`
- Es importante que uséis las opciones de compilación del Jutge de PRO2 (ved Documentation → Compilers → PRO2 a www.jutge.org)
- No usar la opción `-D_GLIBCXX_DEBUG` o usarla de forma incorrecta podrá ser penalizado
- Si no se entrega la documentación generada por el Doxygen, o ésta es incompleta (por ejemplo, si no se pueden ver los elementos privados o la implementación de todas las operaciones), la nota de la corrección manual será cero. Comprobad que vuestra documentación es correcta antes de la entrega. Es necesario usar un Doxyfile basado en el de la sesión 10 de laboratorio para generar correctamente la documentación

Producid el fichero `practica.tar` con la instrucción Linux

```
tar -cvf practica.tar fitxer1 fitxer2 fitxer3 ...
```

desde el directorio/carpeta donde tengáis los ficheros que vais a entregar. Incluid esta instrucción en vuestro Makefile, de forma que el `.tar` se pueda generar ejecutando `make practica.tar`. Con eso reduciréis el riesgo de error en sucesivas entregas. El Jutge no acepta `.tar` donde los ficheros estén dentro de carpetas. Recomendamos usar GNU `tar` para reducir el riesgo de que el fichero `practica.tar` sea incompatible con el Jutge. No es necesario incluir `BinTree.hh` en `practica.tar`.

Ejemplo de entrada 1

<pre> proc1 10 proc11 100 proc111 500 * * proc1 500 * * proc122 500 * * 4 1111 11 11111 111 imprimir_estructura_cluster imprimir_area_espera imprimir_procesadores_cluster alta_proceso_procesador pr 99 15 15 app proc12 99 10 15 imprimir_procesador pr modificar_cluster pp p1 100 * * mc proc1 p2 </pre>	<pre> 100 * 12 500 * * proc12 100 proc121 ipro proc12 mc proc12 p3 100 * * mc proc121 p1 100 p2 100 * * p3 100 * * iec ipc ipro proc12 </pre>
--	---

```
app proc1 99 10 10
ipro proc1
alta_proceso_procesador
proc1 99 5 5
app proc12 101 91 15
app proc12 101 90 15
ipc
baja_proceso_procesador
pr 99
bpp proc12 100
bpp proc12 99
ipro proc12
avanzar_tiempo 7
imprimir_procesador proc12
at 2
ipro proc1
alta_proceso_espera 222 99 10 10

ape 111 99 10 10

ape 111 99 20 15

ape 111 98 1000 20

ape 111 100 10 1

imprimir_prioridad 222
ipri 111
alta_prioridad 1111
ap 2
baja_prioridad 333
bp 111
bp 11
ape 1111 80 10 10
ape 111 80 5 5
iae
enviar_procesos_cluster 1
ipc
iae
epc 100
ipc
iae
configurar_cluster
sndr01
1000
*
*
iec
cc
abc
200
*
zsda
550
*
*

iec

iae
```

```
cmp proc
compactar_memoria_procesador abc

ipro abc
app abc 40 50 8
app abc 30 50 5
app abc 10 50 10
app abc 20 50 5
app zsda 30 50 15
at 5
compactar_memoria_cluster
ipc
at 4
cmc
ipc
fin
```

Ejemplo de salida 1

```
#imprimir_estructura_cluster
(proc1(proc11(proc111 ) (proc112 )) (proc12(proc121 ) (proc122 )))
#imprimir_area_espera
11
0 0
111
0 0
1111
0 0
11111
0 0
#imprimir_procesadores_cluster
proc1
proc11
proc111
proc112
proc12
proc121
proc122
#alta_proceso_procesador pr 99
error: no existe procesador
#app proc12 99
#imprimir_procesador pr
error: no existe procesador
#modificar_cluster pp
error: no existe procesador
#mc proc1
error: procesador con auxiliares
#ipro proc12
0 99 10 15
#mc proc12
error: procesador con procesos
#mc proc121
#iec
(proc1(proc11(proc111 ) (proc112 )) (proc12(proc121 ) (proc122 )))
#ipc
p1
p2
p3
proc1
proc11
proc111
proc112
proc12
0 99 10 15
proc122
#ipro proc12
0 99 10 15
#app proc1 99
#ipro proc1
0 99 10 10
#alta_proceso_procesador proc1 99
error: ya existe proceso
#app proc12 101
error: no cabe proceso
#app proc12 101
#ipc
p1
p2
```

```
p3
proc1
0 99 10 10
proc11
proc111
proc112
proc12
0 99 10 15
10 101 90 15
proc122
#baja_proceso_procesador pr 99
error: no existe procesador
#bpp proc12 100
error: no existe proceso
#bpp proc12 99
#ipro proc12
10 101 90 15
#avanzar_tiempo 7
#imprimir_procesador proc12
10 101 90 8
#at 2
#ipro proc1
0 99 10 1
#alta_proceso_espera 222 99
error: no existe prioridad
#ape 111 99
#ape 111 99
error: ya existe proceso
#ape 111 98
#ape 111 100
#imprimir_prioridad 222
error: no existe prioridad
#ipri 111
99 10 10
98 1000 20
100 10 1
#alta_prioridad 1111
error: ya existe prioridad
#ap 2
#baja_prioridad 333
error: no existe prioridad
#bp 111
error: prioridad con procesos
#bp 11
#ape 1111 80
#ape 111 80
#iae
111
99 10 10
98 1000 20
100 10 1
80 5 5
0 0
1111
80 10 10
0 0
11111
0 0
2
0 0
```

#enviar_procesos_cluster 1	3 1
#ipc	1111
p1	1 0
p2	11111
p3	0 0
procl	2
0 99 10 1	0 0
procl1	#configurar_cluster
procl11	#iec
procl12	(sndr01)
procl2	#cc
0 99 10 10	#iec
10 101 90 6	(abc (zsda))
procl22	#iae
#iae	111
111	98 1000 20
98 1000 20	3 1
100 10 1	1111
80 5 5	1 0
1 0	11111
1111	0 0
80 10 10	2
0 0	0 0
11111	#cmp proc
0 0	error: no existe procesador
2	#compactar_memoria_procesador abc
0 0	#ipro abc
#epc 100	#app abc 40
#ipc	#app abc 30
p1	#app abc 10
0 80 10 10	#app abc 20
p2	#app zsda 30
p3	#at 5
procl	#compactar_memoria_cluster
0 99 10 1	#ipc
procl1	abc
0 100 10 1	0 40 50 3
10 80 5 5	50 10 50 5
procl11	zsda
procl12	0 30 50 10
procl2	#at 4
0 99 10 10	#cmc
10 101 90 6	#ipc
procl22	abc
#iae	0 10 50 1
111	zsda
98 1000 20	0 30 50 6

Información del problema

Autoría: PRO2

Generación: 2026-01-25T21:26:36.789Z

© *Jutge.org*, 2006–2026.

<https://jutge.org>