TERCER LLIURAMENT DE PROP

Grup 10.04

Laura Álvarez Berrocal
Gabriel Chasipanta Gaviño
Manuel Domènech López
Sergi Vilches Llobet

ÍNDEX

MANUAL D'USUARI

Per a què serveix el programa?

El nostre programa serveix com a eina per a crear i jugar kakuros. Així doncs, es tracta d'un programa interactiu on l'usuari pot resoldre kakuros. Aquests kakuros poden haver sigut creats prèviament pel programa o els pot proposar l'usuari. En el cas que els proposi l'usuari, el programa els validarà abans de poder resoldre'ls.

A més a més, consta d'un sistema de rànquing per temps de resolució i per nombre de kakuros resolts per tots els usuaris del sistema.

Tant el repositori com el rànquing es poden reiniciar, això és útil, per exemple, quan es vol fer una competició de kakuros.

Així doncs es tracta d'un programa que no només està enfocat a l'entreteniment individual, sinó que dóna opció a fer competicions.

On és el programa i com s'executa.

El nostre programa es troba dintre de la carpeta EXE. Aquí trobem un arxiu: Executable.jar que únicament caldrà clicar dues vegades per executar.

Informació d'entrada requerida i sortida generada.

Per tal d'iniciar sessió es requereix introduir l'usuari i la contrasenya, això portarà a l'usuari al menú principal.

En cas que es vulgui registrar caldrà introduir un nom d'usuari i una contrasenya que caldrà escriure dues vegades. La resposta del programa serà informar del correcte registre del compte i mostrar la pantalla d'inici de sessió.

Per començar un nou kakuro cal introduir un nombre de files i columnes i una dificultat. A continuació es mostrarà la partida amb el nou kakuro generat.

Tant en visualitzar el repositori de kakuros com en visualitzar partides començades hi haurà d'escollir una de les tres opcions: jugar, eliminar o reiniciar, en cas de repositori de kakuros, o eliminar partides començades, en l'altre cas. En cas que l'opció escollida sigui jugar o eliminar, caldrà escollir un nombre de kakuro vàlid.

En proposar un kakuro l'usuari haurà d'introduir un kakuro en format .txt i el programa l'informarà de la validesa del kakuro i/o de l'identificador d'aquest.

Per tal de visualitzar el rànquing caldrà, primerament, escollir un dels dos tipus i, a continuació, una dificultat. El programa mostrarà una finestra amb el rànquing de la dificultat i el tipus demanat per l'usuari.

Funcionalitats.

Primerament, l'usuari haurà d'iniciar sessió *(Figura 1)*. El nostre programa controla no només que coincideixin l'usuari i la contrasenya amb les dades desades, sinó que els dos paràmetres estiguin omplerts. En cas que un dels dos sigui buit, el nom d'usuari no existeixi o no coincideixin usuari i contrasenya, sortirà un avís com el de la figura *(Figura 2)*.

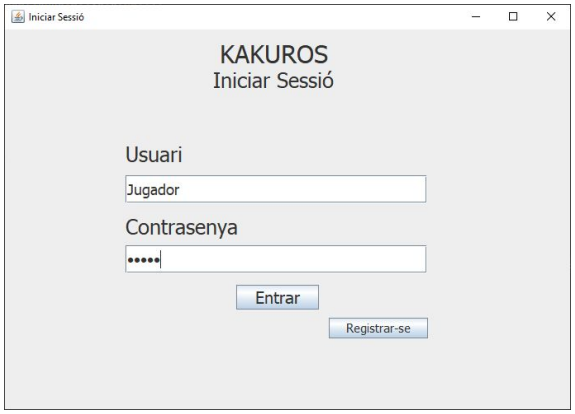


Figura 1: iniciar sessió.

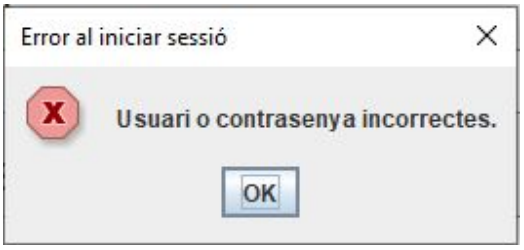


Figura 2: error al iniciar sessió.

Per altra banda, en cas que no tingui compte, haurà de registrar-se clicant el botó de baix a la dreta "Registrar-se". En aquest moment es demanarà introduir un nom d'usuari i una mateixa contrasenya que s'haurà d'escriure dues vegades *(Figura 3)*. En cas que el compte es creï de forma correcta sortirà un avís *(Figura 4)*. Per altra banda, trobarem tres tipus de missatges d'error: un camp buit *(Figura 5)*, les contrasenyes no coincideixen *(Figura 6)* o el nom d'usuari ja està en ús *(Figura 7)*.

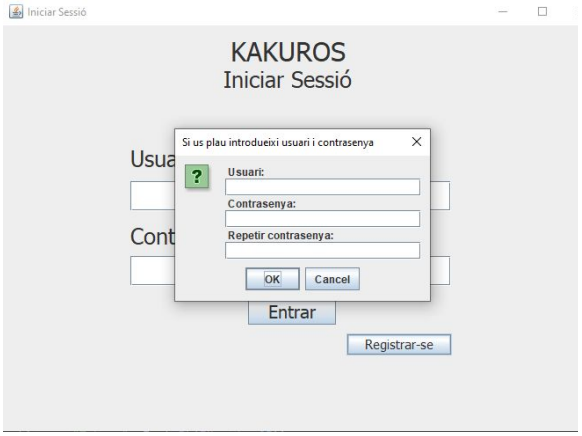


Figura 3: registrar compte.

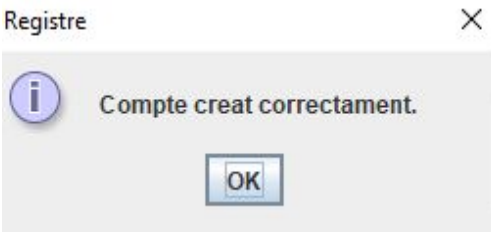


Figura 4: registre correcte.

Figura 5: error al registrar-se 1.

Figura 6: error al registrar-se 2.

Figura 7: error al registrar-se 3.

Un cop registrat, es tornarà a la pantalla d'inici de sessió i l'usuari haurà d'omplir les seves dades. Aleshores entrarem al menú *(Figura 8)*, on trobarem les diferents utilitats del programa i, per cada una, un botó que ens proporcionarà la informació necessària.

Figura 8: menú

A part de l'opció de tancar sessió, la qual ens portarà de volta a la pantalla d'inici *(Figura 1)* trobarem cinc utilitats més:

Primerament trobem nou kakuro: aquesta opció dóna la possibilitat a l'usuari de començar una nova partida. En aquest cas l'usuari haurà d'escollir una mida i una dificultat perquè el programa creï un nou kakuro *(Figura 9)*. A

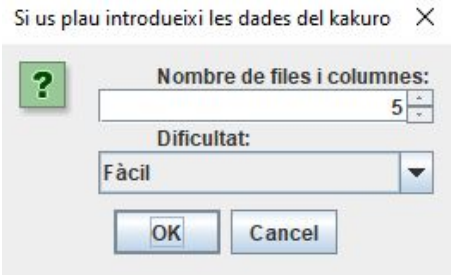


Figura 9: nou kakuro.

continuació ens portarà a la pantalla de partida,
de la qual en parlarem més endavant.

A continuació està Repositori *(Figura 10)*. Al repositori trobarem els identificadors del kakuros desats i ens deixarà escollir entre tres opcions: jugar, que ens portarà a partida; eliminar, que eliminarà de forma permanent el kakuro del nostre repositori; i reiniciar, que eliminarà tot el repositori de kakuros de forma definitiva. Tant per eliminar kakuro com per reiniciar el repositori el programa demanarà confirmació a l'usuari ja que és una acció irreversible *(Figures 11 i 12)*. En cas que l'usuari intenti jugar o eliminar, però no seleccioni un kakuro, el sistema li informarà que ha de seleccionar un identificador de kakuro *(Figura 13)*.

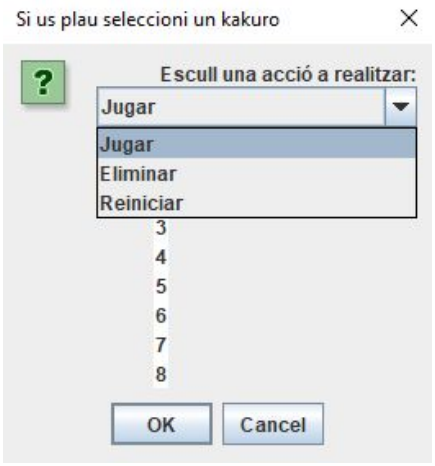


Figura 10: repositori de kakuros.

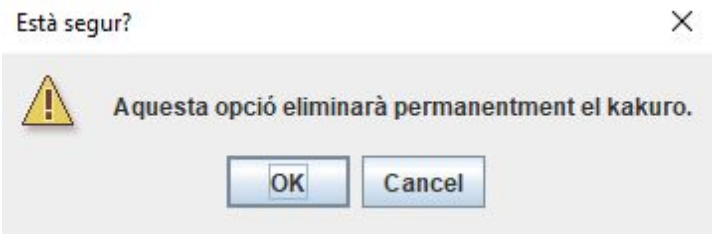


Figura 11: eliminar kakuro del repositori.

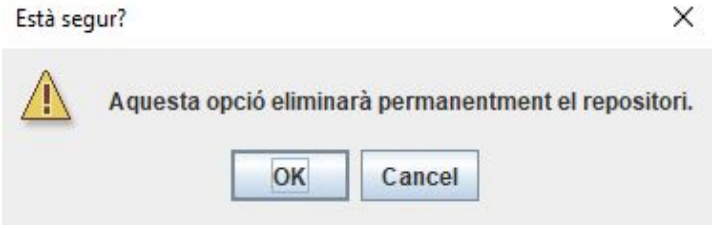


Figura 12: reiniciar repositori de kakuros.

FIgura 13: identificador no seleccionat.

Si l'usuari escull proposar kakuro, s'obrirà una finestra per escollir un document amb extensió .txt *(Figura 14)*. El document haurà de tenir un kakuro en el format demanat, del qual es troba la informació necessària a la dreta de la icona de proposar kakuro del menú. En cas que no es seleccioni cap arxiu sortirà un missatge d'error *(Figura 15)*. Un cop pujat l'arxiu, el sistema comprovarà que el kakuro sigui vàlid. En cas que ho sigui es preguntarà a l'usuari si vol jugar o guardar el kakuro al repositori *(Figura 16),* cas en el qual es retornarà l'identificador del

kakuro *(Figura 17)* amb el qual el podrà troba al repositori. Per altra banda, si el kakuro no és vàlid s'informarà l'usuari *(Figura 18)*.

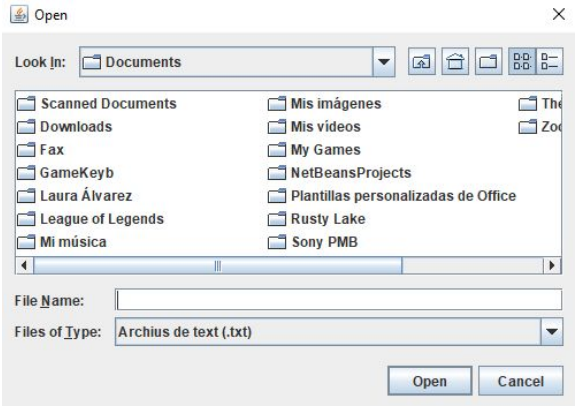


Figura 14: pujar kakuro.

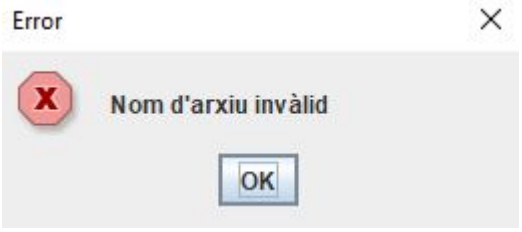


Figura 15: error al pujar arxiu.

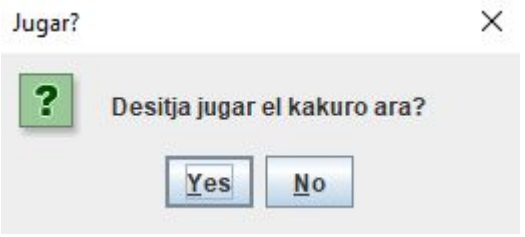


Figura 16: desitja jugar?

Figura 17: es retorna l'id del kakuro validat.

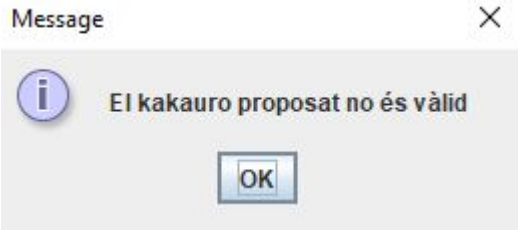


Figura 18: el kakuro no és vàlid.

Tot seguit trobem l'opció: Partides començades *(Figura 19)*. Aquesta permet a l'usuari visualitzar les partides que s'han deixat a mitges. En aquest punt en trobem 3 opcions: jugar, que permet a l'usuari obrir un kakuro ja començat; eliminar, que esborra de forma permanent una partida del repositori; i eliminar partides a mitges, que dóna l'opció d'eliminar tot el repositori de partides començades. Tant per eliminar una partida *(Figura 20)* com eliminar el repositori *(Figura 21)* es demanarà confirmació, ja que és irreversible. En cas que es seleccioni jugar o eliminar partida, caldrà seleccionar un identificador de partida de la llista, si no es fa s'informarà a l'usuari *(Figura 22)*.

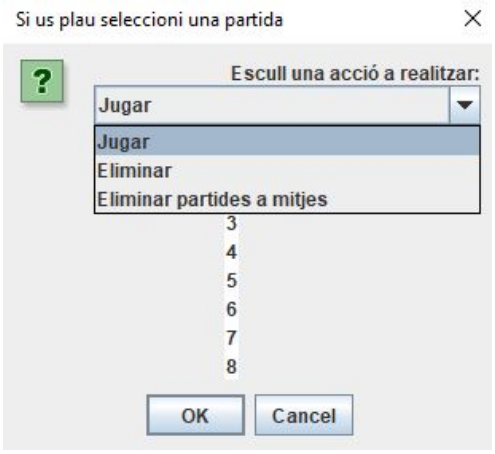


Figura 19: partides començades.

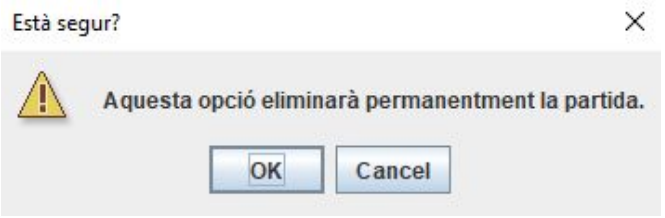


Figura 20: eliminar partida.

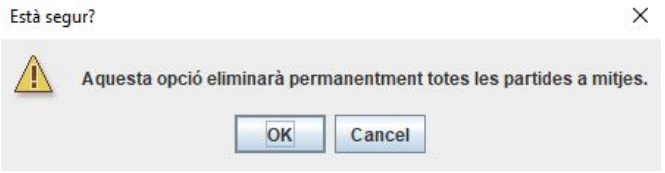


Figura 21: eliminar partides a mitges.

Figura 22: identificador no seleccionat.

Qualsevol de les opcions anteriors dóna la possibilitat de començar una partida. En aquest cas visualitzarem la pantalla partida *(Figura 23)*. Primerament, sobre el kakuro, podem veure dos dígits: el número d'identificador de la partida i el temps emprat fins al moment en la resolució del kakuro. A més, podem trobar diversos botons: Enrere, Màquina, Ajuda, Pista i Comprova.

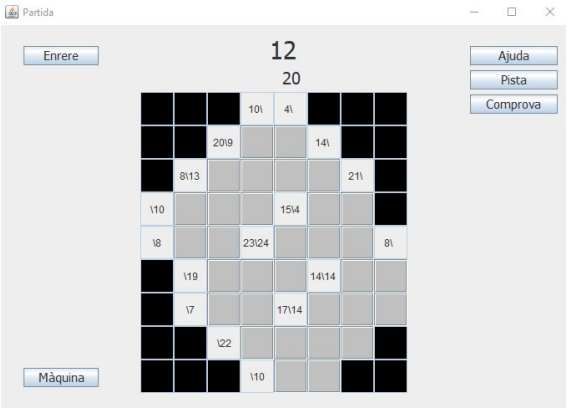


Figura 23: partida.

Enrere permet a l'usuari tornar al menú. En el moment en què l'usuari premi el botó la partida es guardarà automàticament per tal de poder jugar-la quan es desitgi.

Màquina informa l'usuari del temps que ha emprat el sistema en la resolució del kakuro *(Figura 24)*.

Figura 24: temps de resolució de la màquina.

El botó Ajuda mostra l'usuari les normes per realitzar kakuros *(Figura 25)*.

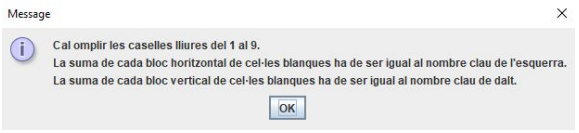


Figura 25: Ajuda.

En prémer Pista el programa ens facilitarà la solució d'una de les cel·les blanques del kakuro que encara no hagin estat omplertes *(Figura 26)*.

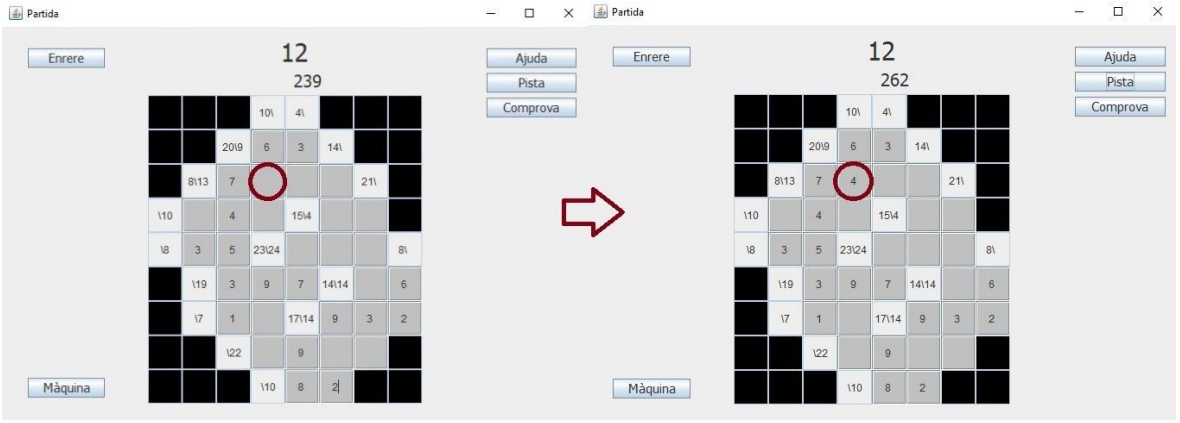


Figura 26: Pista.

Finalment, trobem Comprovar. Aquest serveix per saber si totes les cel·les omplertes tenen el nombre que pertany a l'única solució correcta del kakuro. En cas que el nombre sigui erroni, la cel·la es pintarà de vermell *(Figura 27)*. A més, un cop acabat el kakuro, l'usuari haurà de prémer aquest botó per saber si la seva solució és correcta *(Figura 28)*.

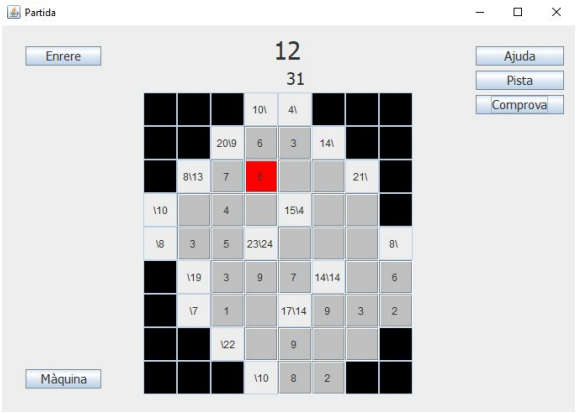


Figura 27: Comprova.

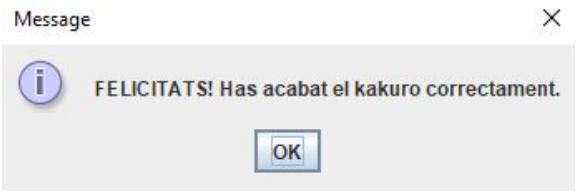


Figura 28: Comprova al finalitzar el kakuro.

La darrera opció al menú es rànquing. En aquest cas es portarà a l'usuari a una pantalla on podrà escollir quin tipus de rànquing vol mostrar *(Figura 29)*: per temps mitjà o per nombre de kakuros resolts. En qualsevol dels dos casos, l'usuari haurà

d'especificar quina dificultat li interessa veure *(Figura 30)* i, a continuació, es mostrarà el rànquing *(Figura 31)*.

A més a més, tenim l'opció de Reiniciar. Aquesta reinicia tot el rànquing i, com és una acció irreversible, demana a l'usuari confirmació *(Figura 32)*.

Figura 29: menú de rànquing.

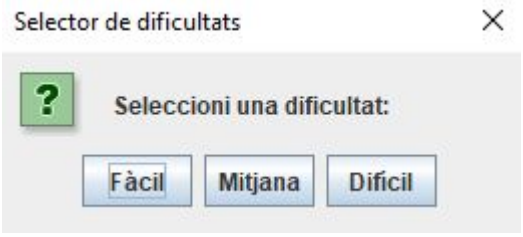


Figura 30: seleccionar dificultat.

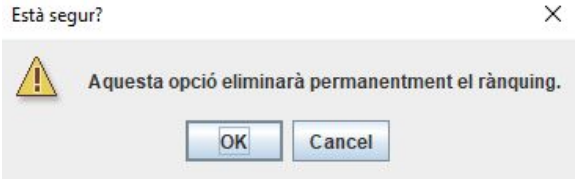
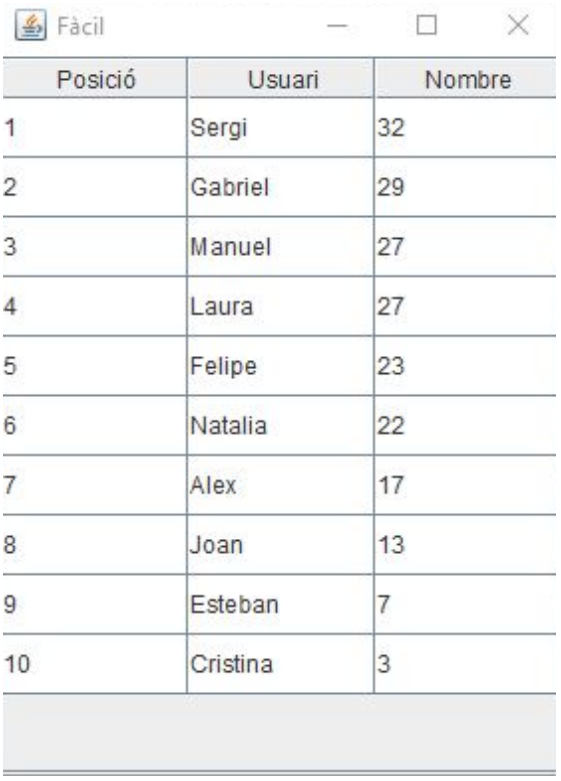


Figura 32: reiniciar rànquing.



Posició	Usuari	Nombre
1	Sergi	32
2	Gabriel	29
3	Manuel	27
4	Laura	27
5	Felipe	23
6	Natalia	22
7	Alex	17
8	Joan	13
9	Esteban	7
10	Cristina	3

Figura 31: rànquing.

Exemples d'utilització.

- L'usuari vol jugar un kakuro que tenim guardat:

  En aquest cas caldrà, un cop al menú, clicar a "Repositori de kakuros". A continuació es mostrarà una finestra que l'usuari haurà d'escollir que vol fer *(Figura 10)*. L'usuari escull un identificador de kakuro dels quals disposa al repositori i escull l'acció de jugar. Aleshores el programa el portarà a la partida, on es veurà el kakuro amb les diferents utilitats.

- L'usuari vol continuar una partida ja començada:

L'usuari haurà d'escollir l'opció "Partides començades" on haurà d'escollir un dels identificadors de partida i escollir l'opció de jugar *(Figura 19)*, després es mostrarà la partida amb el kakuro i el temps acumulat en la resolució.

- L'usuari vol visualitzar el rànquing per nombre de kakuros resolts en dificultat difícil:

  En aquest cas caldrà accedir a "rànquing" i, a continuació, a "Per nombre" *(Figura 29)*. En aquest moment el programa demanarà a l'usuari escollir una dificultat, en aquest cas "Difícil" *(Figura 30)*. Aleshores es mostrarà per pantalla el rànquing demanat.

- L'usuari vol proposar un kakuro:

  Després de clicar a "Proposar kakuro" s'obrirà una finestra per escollit un fitxer .txt amb un kakuro fet per l'usuari *(Figura 14)*. El programa el validarà i, en cas que no vulgui jugar-lo, se li retornarà un identificador de kakuro per tal de poder jugar-lo més tard buscant-lo al repositori.

  Un exemple del format de proposta de kakuro de l'usuari podria ser:

  8,8
  ,,C24,C15,*,*,C17,C3
  ,C3F10,?,?,,C10F10,?,?
  F11,?,?,?,C24F11,?,?,?
  F25,?,?,?,?,?,*,*
  ,,C6F14,?,?,?,*,*
  *,C17F16,?,?,?,?,C17,C3
  F11,?,?,*,*,F10,?,?
  F11,?,?,*,*,F10,?,?
  mitjana

- L'usuari vol comprobar si el kakuro resolt és correcte:

  En cas que l'usuari hagi completat un kakuro i vulgui saber si la seva solució és correcta, haurà de, dintre de partida, clicar a "Comprovar". En aquest punt, si la solució és correcte s'informarà l'usuari*(Figura 28)*; en cas contrari, es marcaran en vermell les cel·les amb valor incorrecte*(Figura 27)*.

- L’usuari vol reiniciar el rànquing de kakuros:

  En cas que l'usuari per qualsevol motiu, com bé podria ser una competició de kakuros, vulgui reiniciar tot el rànquing, haurà de dirigirse a "rànquing" i un cop allà, haurà d'escollir l'opció de "Reiniciar" *(Figura 29)*. Aleshores sortirà un missatge per assegurar l'acció, moment en el qual l'usuari ha de donar a "Ok" *(Figura 32)*.

- L’usuari vol començar a jugar un kakuro nou:

  Per tal de començar una partida amb un nou kakuro, l’usuari haurà de clicar a “Nou kakuro”. Un usuari que està aprenent a fer kakuros, per exemple, podria crear un kakuro senzill, això és un kakuro de per exemple 5x5 amb dificultat fàcil *(Figura 9)*. En qualsevol cas, es mostrarà la pantalla de partida amb el kakuro creat.

- L’usuari vol guardar un kakuro començat per jugar-lo més tard:

  En aquest cas, l’usuari que es troba a Partida *(Figura 23)* hi haurà de prémer el botó Enrere, en aquest moment es tornarà a l’usuari a la pantalla de menú i la partida es guardarà automàticament.

Capacitats i limitacions.

Capacitats:

El programa genera kakuros quadrats té diverses limitacions i capacitats. Primerament, pot arribar a generar Kakuros de 9x9 en nivell difícil en un temps aproximat de 1s, el qual el fa que sigui molt eficient. Els kakuros generats els resol el nostre resolutor de kakuros. A més, permet a l'usuari proposar kakuros que seran validats.

Una gran capacitat de l'algorisme de resolució és la seva velocitat. Els seus temps són bastant acceptables (menys d’un segon en algorismes 9*9) i algo més en kakuros més complexes i la seva validació també ho és. En conjunt fa que sigui un bloc compacte i eficient.

Permet a l'usuari començar una partida ràpida generant un nou kakuro, el qual es guardarà a partides començades. També pot escollir un kakuro dels ja generats, els

quals estaran desats al repositori de kakuros; o continuar amb una partida inacabada.

Pel que fa a la jugabilitat, controla el temps que triga un usuari a realitzar un kakuro i, en cas que deixi la partida a mitges per tornar més endavant, guarda el temps emprat fins al moment. Permet a l'usuari comprovar el kakuro en qualsevol moment, demanar pistes i informar-se del temps emprat per la màquina en resoldre el mateix kakuro. Finalment, l'usuari pot, en qualsevol moment, visualitzar les normes per resoldre kakuros.

L'usuari té l'opció de visualitzar dos tipus de rànquing. Un per nombre de kakuros resolts per cada dificultat, i un altre per temps mitjà emprat en resoldre kakuros de cada dificultat. Tant un com l'altre s'actualitzen cada cop que l'usuari acaba un kakuro.

El repositori de kakuros, les partides començades i el rànquing donen l'opció a l'usuari de ser reiniciats. Tanmateix, aquesta acció és irreversible.

Limitacions:

Sí és cert que no té la la certesa total de generar kakuros vàlids per si mateix, per tant el que fem és passar el resolutor (algorisme de validació) per veure si el kakuro generat té més d'una solució.

Respecte al resolutor té la limitació del seu ús de memoria. El kakuro conté matrius estàtiques que ocupen al voltant de 3000 elements i per tant fa que ocupi una gran part d'aquesta. També cal remarcar que a l'hora és una gran virtud ja que aquests valors no els ha de computar a cada iteració i això el converteix en un algorisme encara més eficient.

El programa només genera kakuros quadrats, tanmateix, en resol de tots els tipus. Així doncs l'usuari només pot demanar partides de kakuros quadrats però en podrà proposar que no en siguin.

Respecte a la partida, no compta amb un monitoratge en temps real, així doncs, haurà de demanar una comprovació quan vulgui saber si és correcte.

Canvis de última hora.

La classe Partida que a la segona entrega tenia una relació d'agregació amb les classes kakuro i usuari l'hem canviat a una relació d'associació. Aquest ha estat un canvi no esperat de última hora que ens hem plantejat fer per a millorar la eficiencia del codi i de l'espai de memòria.

Algunes de les funcions de la classe Controlador_Persistencia han estat eliminades, ja que les hem inclòs en d'altres. A més a més, trobem alguna de la qual hem canviat el nom per aclarir-ne l'ús.

Hem implementat una segona versió de l'algorisme de resolució de kakuros. Aquesta comprova de forma inversa els números que poden omplir les caselles blanques per a trobar una solució. Essencialment, ens ha ajudat a validar si un kakuro és vàlid, ja que és capaç de trobar una solució diferent, si hi ha més d'una, a la que trobaria la primera versió de l'algorisme.

Hem afegit la funció "public ArrayList<String> guardar()" a la classe Ranking que s'encarrega de guardar en un vector les dades del Ranking tal com s'hauran d'escriure al Ranking.txt.

Al Controlador_Domini li hem afegit 2 funcions també. "public void CD_reiniciar_rànquing()" s'encarrega d'esborrar el Ranking que hi havia i posar-ne un de nou buit. "public void CD_add_rank(String us, String ds, int t)" afegeix la partida al Ranking quan aquesta finalitza.

Al Controlador_Persistencia li hem canviat la funció "del()" per "CP_restart_rank()" que elimina el Ranking antic i en posa un de nou buit, en lloc de només eliminar.

JOCS DE PROVA

PROVA1:

Descripció: Generació de kakuro 4*4 (el més petit possible).

Objectius: Buscar si l'algorisme de generació conté problemes en la creació de kakuros petits.

Entrada: 4 files i 4 columnes dificultat fàcil.

Sortida: L'algorisme genera el kakuro de forma correcte i no es queda el programa encallat.

Resultat de la prova: OK.

PROVA 2:

Descripció: Generació de kakuro 9*9 (el més gran possible).

Objectius: Buscar si l'algorisme de generació conté problemes en la creació de kakuros grans.

Entrada: 9 files i 9 columnes dificultat difícil.

Sortida: L'algorisme genera el kakuro de forma correcte i no es queda el programa encallat.

Resultat de la prova: OK.

PROVA 3:

Descripció: Guardar una partida.

Objectius: Mirar si les partides es creen i es guarden de forma correcte amb el seu corresponent identificador.

Entrada: Generar un kakuro qualsevol i guardar la partida a mitges.

Sortida: Al guardar la partida a mitges ens diu que tenim l'identificador de la nostra partida a 0. Anem a partides existents i podem comprovar que la tenim i que és la partida correcta.

Resultat de la prova: OK.

PROVA 4:

Descripció: Seleccionar un kakuro del repositori i jugar-lo.

Objectius: Mirar si els kakuros estan guardats de forma correcta al repositori i començar una partida.

Entrada: Seleccionar un kakuro del repositori i seleccionar l'opció de jugar.

Sortida: La partida és creada de forma correcte.

Resultat de la prova: OK.

PROVA 5:

Descripció: Seleccionar un kakuro del repositori i eliminar-lo.

Objectius: Mirar si els kakuros estan guardats de forma correcta al repositori i si els podem eliminar.

Entrada: Seleccionar un kakuro del repositori i seleccionar l'opció d'eliminar.

Sortida: Ens surt un missatge d'avís informant-nos de la situació i un cop acceptat podem recarregar els kakuros disponibles que tenim a la app. Podem apreciar que el kakuro anterior ja no forma part del repositori.

Resultat de la prova: OK.

PROVA 6:

Descripció: Eliminar tots els kakuros disponibles del repositori.

Objectius: Mirar si els kakuros estan guardats de forma correcta al repositori i si els podem eliminar en complet.

Entrada: Seleccionar l'opció d'eliminar tots els kakuros.

Sortida: Ens surt un missatge d'avís informant-nos de la situació i un cop acceptat podem recarregar els kakuros disponibles que tenim a la app. Podem apreciar que el repositori ja és buit.

Resultat de la prova: OK.

PROVA 7:

Descripció: Proposta Kakuro vàlid.

Objectius: Mirar si l'algorisme de validació de propostes de kakuros funciona de forma correcta.

Entrada: Un arxiu de text amb un kakuro vàlid en format estàndard.

Sortida: El kakuro és guardat al repositori de forma correcte.

Resultat de la prova: OK.

Kakuro proposat:

```
5,5
*,*,*,C7,C6
*,C7,C24F6,?,?
F15,?,?,?,?
F13,?,?,?,?
F10,?,?,*,*
facil
```

PROVA 8:

Descripció: Proposta Kakuro no valid.

Objectius: Mirar si l'algorisme de validació de propostes de kakuros funciona de forma correcta.

Entrada: Un arxiu de text amb un kakuro no vàlid en format estàndard.

Sortida: Salta un avís d'error conforme ens indica que aquest kakuro no és correcte.

Resultat de la prova: OK.

Proposta kakuro:

```
5,5
*,C10,C10,*,*
F3,?,?,*,*
F17,?,?,C10,C10
*,*,F17,?,?
*,*,F3,?,?
mitjana
```

PROVA 9:

Descripció: Comprovació del registre d'usuaris.

Objectius: Mirar si el registre d'usuaris es manté correcte.

Entrada: Anem a la part del registre i introduïm un nou nom d'usuari (vàlid) amb la seva contrasenya. Posteriorment intentem "loggin" amb un nom d'usuari diferent i una contrasenya diferent a la proposada. Seguidament proposem l'usuari que hem indicat anteriorment amb una contrasenya incorrecta i finalment proposem tot en com ho haviem posat al registre anterior.

Sortida: Salta un avís d'error indicant que l'usuari o la contrasenya són incorrectes en els primers dos casos i en el tercer entra al programa.

Resultat de la prova: OK.

PROVA 10:

Descripció: Comprovació del "loggin" d'usuaris.

Objectius: Mirar si el "loggin" d'usuaris es manté correcte.

Entrada: Anem a la part del registre i introduïm un nou nom d'usuari no vàlid (ja registrat anteriorment) i una proposta de contrasenya.

Sortida: Salta un avís d'error indicant que el nom d'usuari ja havia estat registrat anteriorment.

Resultat de la prova: OK.

PROVA 11:

Descripció: Comprovació del "mostra rànquing per nombre".

Objectius: Mirar si el Ranking per nombre es mostra correctament en cadascuna de les dificultats.

Entrada: Acabem al menys 2 partides de cada dificultat per si el rànquing estigués buit. Tot seguit anem a l'apartat de rànquing i demanem que ens mostri el rànquing per nombre de cada dificultat.

Sortida: Observem que ens mostra el rànquing per nombre de cada dificultat correctament.

Resultat de la prova: OK.

PROVA 12:

Descripció: Comprovació del "mostra rànquing per temps".

Objectius: Mirar si el Ranking per temps es mostra correctament en cadascuna de les dificultats.

Entrada: Acabem al menys 2 partides de cada dificultat per si el rànquing estigués buit. Tot seguit anem a l'apartat de rànquing i demanem que ens mostri el rànquing per temps de cada dificultat.

Sortida: Observem que ens mostra el rànquing per temps de cada dificultat correctament.

Resultat de la prova: OK.

PROVA 13:

Descripció: Comprovació del “add” d’una partida al rànquing.

Objectius: Mirar si l’“add” de la partida s’executa correctament.

Entrada: Observem el rànquing actual, juguem un nou Kakuro, l’acabem i finalitzem la partida. Anem a l’apartat de rànquing i l’observem altre cop.

Sortida: Observem que la partida s’ha afegit correctament.

Resultat de la prova: OK.

PROVA 14:

Descripció: Comprovació del “Reiniciar” del rànquing.

Objectius: Mirar si el “Rreiniciar” del rànquing s’executa correctament.

Entrada: Observem el rànquing actual i si està buit, juguem un nou Kakuro, l’acabem i finalitzem la partida. Tornem a observar el rànquing i veiem que hi ha dades guardades. Cliquem a reiniciar i tornem a mirar el rànquing.

Sortida: Observem que el rànquing està buit.

Resultat de la prova: OK.

PROVA 15:

Descripció: Comprovació de la correctesa de la diversitat d’usuaris a les partides disponibles.

Objectius: Comprovar si realment cada usuari té les seves pròpies partides disponibles i altres usuaris no hi poden accedir a elles.

Entrada: Ens registrem amb un usuari qualsevol, posteriorment podem crear una partida de qualsevol forma de les 3 vàlides (Generant un kakuro, proposant un kakuro o sel·leccionant un kakuro del repositori) i finalment deixem la partida a mitges. Un cop tenim tot això fet comprovem que tenim una partida disponible amb el identificador que se li ha assignat anteriorment. Seguidament, ens registrem amb un altra usuari i anem a l'apartat de partides disponibles. Podem observar que és buit.

Sortida: Mirant que les partides disponibles del segon usuari són nules aleshores podem verificar que els usuaris no comparteixen partides a mitges.

Resultat de la prova: OK.

CLASSES IMPLEMENTADES

La classe Kakuro ha estat realitzada per Gabriel; a excepció dels algorismes. L'algorisme resolutor ha estat fet per Gabriel i Laura, del qual en trobem dues variants més, una per validar, realitzada per ambdós membres, i una per calcular el temps de resolució, feta per Gabriel. La versió definitiva de l'algorisme de generació ha estat implementada completament per Sergi aprofitant la versió de la primera entrega en la qual va participar el Manuel. El Repositori_Kakuros ha estat fet per Sergi.

La classe Partida ha estat feta per Sergi amb la intervenció de Gabriel. El Repositori_Partides ha estat fet per Sergi.

Quadrat ha estat implementada pels quatre membres del grup.

Ranking ha estat realitzada per Manuel i Laura.

La classe Usuari ha estat escrita per Manuel i debugada per Sergi. La classe Repositori Usuaris ha estat escrita per Sergi i Manuel i debugada per Sergi.

Gestor_Ranking ha estat realitzada per Manuel i Gestor_Usuaris ha estat escrita per Sergi i Manuel i debugada per Sergi. Mentre que les altres tres classes gestores (Gestor_Kakuro i Gestor_Partides) han estat fetes per Sergi.

Les vistes de presentació han estat realitzades per Laura.

El Controlador_Presentacio ha estat realitzat per Laura amb l'ajuda de Gabriel. Els altres dos controladors (Controlador_Domini i Controlador_Persistencia) han estat realitzats pels altres tres membres (Gabriel, Sergi i Manuel) a mesura que s'implementaven les classes.

Tenint en compte la distribució anterior, cal considerar alguns aspectes importants. Primerament, la distribució de la càrrega de feina no ha estat equitativa i hi ha hagut parts del projecte que han recaigut en un sol membre quan no hauria d'haver estat així. El 90% del debugament del programa ha estat fet únicament per Sergi, Gabriel i Laura.

Resumint, creiem que el temps dedicat en el projecte no es veu realment reflexat en el nombre de classes realitzades.

LLIBRERIES EXTERNES

Hem utilitzat la llibreria javafx.util per tal de poder fer ús de l'estructura de dades Pair i la llibreria javax.swing per tal de fer la interfície gràfica.