Memory

## هدف:

هدف از این کار این است که به ما بیاموزد که به طور مستقل برنامه هایی در سبک Object Oriented ایجاد کنیم. شما همچنین در گروه های 3-4 نفره کار خواهید کرد و یاد خواهید گرفت که از git brancher استفاده کنید تا کار خود را در بین خود سازماندهی کنید.

## اهداف دوره تحت پوشش تکالیف

# دانش:

* از روش های اثبات شده برای اشکال زدایی کد استفاده کنید
* از کنترل نسخه با Git در ترمینال و GitHub استفاده کنید
* به طور مستقل نرم افزار شی گرا و مطابق با تمرین خوب برنامه نویسی توسعه دهید.

# چیزی که قرار است تحویل دهید

* README.md (با دستورالعمل هایی در مورد تدوین و رانندگی / تست)
* کارت.جاوا
* Player.java
* Game.java
* Visibility.java
* BoardView.java
* UML-diagram.png (تصویر روی نمودار شما)
* Planning.txt (توضیح تفکیک مشکل)
* .git

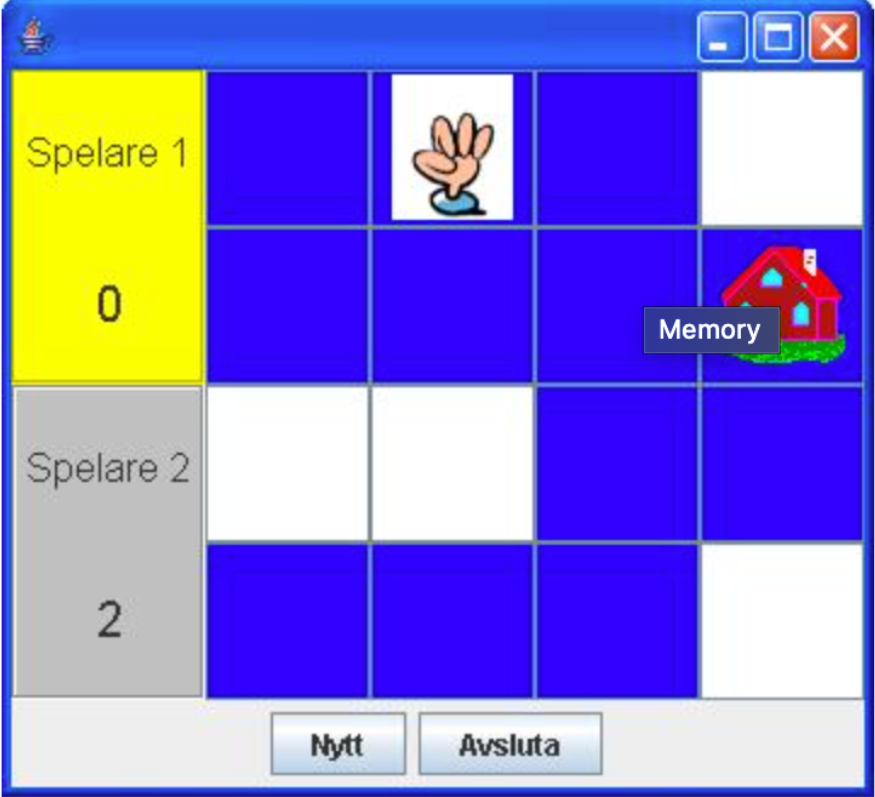
# وظیفه

همانطور که می دانید بازی Memory توسط دو یا چند بازیکن قابل انجام است. این بازی شامل تعدادی کارت با پشت یکسان اما با تصاویر متفاوت در جلو است. هر تصویر دقیقاً روی دو کارت ظاهر می شود. وقتی بازی شروع می شود، تمام کارت ها رو به بالا روی میز قرار می گیرند. سپس بازیکنان باید به نوبه خود دو کارت را برگردانند. اگر دو کارت تصاویر متفاوتی داشته باشند، پس از 2 ثانیه به عقب برگردانده می شوند تا پشت آن رو به بالا باشد. در این صورت، بازیکن بعدی ممکن است ادامه دهد. از طرف دیگر، اگر دو کارت تصویر یکسانی داشته باشند، بازیکن ممکن است کارت‌ها را از روی میز برداشته و در انبوه خود قرار دهد. همان بازیکن می تواند ادامه دهد و دو کارت جدید را برگرداند. یک بازیکن خاص ممکن است تا زمانی که بتواند جفت کارت هایی با همان تصویر را پیدا کند، ادامه دهد. بازی ادامه می یابد تا زمانی که تمام کارت های روی میز از بین بروند. بازیکنی که بیشترین جفت را داشته باشد برنده شده است.

## مشخصات مورد نیاز

وظیفه نوشتن یک برنامه کامپیوتری است که به کاربران اجازه می دهد حافظه را بازی کنند. به جای داشتن کارت روی میز، کارت ها باید روی صفحه نمایش داده شوند. بازیکنان با کلیک بر روی کارت ها کارت ها را انتخاب می کنند. برنامه باید دارای یک پنجره با تعدادی کارت باشد. کارت ها باید در یک شبکه قرار داده شوند. برنامه باید طوری طراحی شود که هنگام شروع برنامه بتوان تعداد سطرها و ستون ها را انتخاب کرد. در پنجره نیز باید دو دکمه وجود داشته باشد که بازیکنان می توانند زمانی که می خواهند یک بازی جدید را شروع کنند یا برنامه را پایان دهند، روی آنها کلیک کنند. برنامه باید دو بازیکن در نسخه اصلی داشته باشد. امتیاز فعلی دو بازیکن، یعنی. تعداد جفت کارت هایی که بازیکنان پیدا کرده اند باید نمایش داده شود. برنامه باید ردیابی کند که نوبت کدام بازیکن است و این باید به وضوح در پنجره مشخص شود. یکی می تواند به عنوان مثال از رنگ های مختلف برای نشان دادن امتیاز بازیکنان استفاده کنید.

نمونه ای از آنچه ممکن است به نظر برسد در شکل زیر نشان داده شده است که در آن بازیکن 1 به تازگی دو کارت را انتخاب کرده است که مشخص شده است متفاوت هستند. در این شکل از رنگ زمینه زرد برای نشان دادن نوبت به چه کسی استفاده شده است. جعبه های سفید نشان می دهد که بازیکن دوم، 2 جفت پیدا کرده است.



شما در نهایت کلاس های زیر را خواهید داشت: کارت، بازیکن، بازی، نمایش صفحه، مشاهده (تعدادی)

* قابلیت مشاهده (تعدادی)

نمایانگر نمایان شدن یک کارت است: VISIBLE، INVISIBLE، ELIMINATED (بازیکنان جفت پیدا کرده اند)

* کارت

نشان دهنده کارت / تصویری است که می تواند تبدیل شود (Visibility) و قابل مشاهده، نامرئی، حذف شود

* بازیکن

نشان دهنده بازیکنی است که می تواند فعال یا غیر فعال (بولی)، دارای امتیاز (int)، نام (رشته) باشد.

* بازی

منطق بازی را نشان می‌دهد و نوبت کدام بازیکن، کارت‌های موجود را دنبال می‌کند و BoardView را کنترل می‌کند. این کلاس نیز برنامه اصلی است که متد اصلی را دارد.

* BoardView

رابط گرافیکی را نشان می‌دهد و کارت‌ها را می‌کشد، بازیکن فعال را علامت‌گذاری می‌کند، و هنگامی که شخصی کارتی را فشار می‌دهد، کارت‌ها را برمی‌گرداند، ترجیحاً به عنوان زیر کلاس JButton نشان داده شود.

برای طراحی بازی، باید یک نمودار UML از کلاس های خود و همچنین نحوه تعامل آنها ایجاد کنید (نمودار توالی). برای این کار می توانید به عنوان مثال از https://app.diagrams.net/ استفاده کنید.

شما همچنین باید کار خود را در این کار برنامه ریزی کنید (ترجیحا قبل از شروع کدنویسی). تفکیک مشکل باید در یک فایل متنی Planning.txt مستند شود. این می تواند فرمت زیر را داشته باشد (مثال ATM):

## خرابی مشکل

1. Bankomat GUI
   1. Rita ut knappar för pinkod, enter och avbryt
   2. Koppla knappar till lyssnare för att låta programmet få input
   3. Printa ut alla intryckta värden (input) vid varje nytt tryck
2. Autentisering
   1. Gör koppling till databas
   2. Fråga efter användare i databas (hämta lösenord)
   3. Kontrollera att användaren har tryckt rätt lösenord (matchar databasen)
3. Fel pin
   1. Om användaren har tryckt fel pin, låt hen göra 2 nya försök
   2. Annars ge felmeddelande
   3. Ge inte tillbaka kort till användaren (stulet)
4. Rätt pin
   1. Om användaren har tryckt rätt pin, visa saldo
   2. Ge användaren val att ta ut pengar
   3. Ta input från användaren om önskat belopp
   4. Kontrollera att användaren inte överskrider sitt saldo
5. Överskrider saldo
   1. Ge information om att användaren inte har tillräckligt på sitt konto
   2. Fråga efter input om nytt önskat belopp att ta ut
6. Har tillräckligt saldo
   1. Gör koppling till databas
   2. Uppdatera användares saldo
   3. Ge användare pengar
   4. Ge kort tillbaka till användare

# الزامات درجه بندی

## رتبه G

برنامه باید دارای مشخصات باشد

پروژه باید با نسخه مدیریت شده با git، با چندین commit که توسعه برنامه را توصیف می کند (توجه: یک commit به عنوان تقلب طبقه بندی می شود و IG می دهد)

## رتبه بندی VG

این پروژه دارای یک صنعت گرید برای هر نفر است که در چندین commit نشان می دهد که هر کدام در پروژه چه کاری انجام داده اند

این پروژه دارای بیش از 5 تست واحد (Junit) برای منطق بازی در بازی است

نمودار طراحی UML

یک فایل متنی (Planning.txt) با توضیحی در مورد اینکه چگونه مشکل را حل کرده اید.

jpannelPlayer1{

backgroundActive

backgroundInactive

}

==========================