**Game Documentation**

**create by**

**Thammasorn Thammasarangkoon 6432072021**

**Peeraphat Wongsri 6430274121**

**2110215 Programming Methodology Semester 1 Year 2022 Chulalongkorn University**

**Casino Royal Dice**

**Introduction**

Casino Royal Dice เป็นเกมที่ได้รับแรงบันดาลใจมากจาก Board game ที่ชื่อว่า “**Las Vegas”** ซึ่งเป็น Board game

ที่เล่นได้ 2-5คน ซึ่งเป้าหมายของเกมคือให้ผู้เล่นสะสมเงินจากในแต่ละสถานที่ให้ได้มากที่สุดใน 4 รอบ

**Rules**

ในแต่ละรอบผู้เล่นจะมีลูกเต๋าในตอนเริ่มเกม 8 ลูก และในแต่ละสถานที่จะมีธนบัตรที่มีมูลค่ารวมกันไม่ต่ำกว่า 50,000 $ ผู้เล่นจะต้องทอยลูกเต๋าทั้งหมดที่ตัวเองมีทุกครั้ง เมื่อทอยเสร็จผู้เล่นจะต้องทำการเลือกลูกเต๋าจำนวนหนึ่งไปวางบนหนึ่งสถานที่จากทั้งหมด 6 สถานที่ ซึ่งแต่ละสถานที่ก็จะมีหน้าลูกเต๋ากำกับอยู่ว่าต้องการลูกเต๋าหน้าหมายเลขใด เช่น สถานที่ A ถูกกำกับด้วยหน้าลูกเต๋าหมายเลข 4 ผู้เล่นก็จะสามารถวางลูกเต๋าหน้าหมายเลข 4 บนสถานที่A ได้เท่านั้น รับประกันว่า ในแต่ละสถานที่จะถูกกำกับด้วยหน้าลูกเต๋า 1 หมายเลข และอยู่ระหว่าง 1-6 เท่านั้น ในแต่ละรอบจะจบก็ต่อเมื่อผู้เล่นทุกคนไม่เหลือลูกเต๋าที่ตัวผู้เล่นแล้วหรือพูดได้ว่าวางลูกเต๋าลงไปทั้งหมดแล้ว เมื่อจบรอบจะพิจารณาดังนี้

1. แจกธนบัตรในแต่ละสถานที่ให้แก่ ผู้เล่น โดยเรียงลำดับการรับเงินจากจำนวนลูกเต๋าที่วางอยู่บนสถานที่นั้น และเริ่มแจกจากธนบัตรใบใหญ่ที่สุดก่อนเสมอ และ ถ้าหากจำนวนธนบัตร มีน้อยกว่า จำนวนผู้เล่นที่วางลูกเต๋า ผู้เล่นจำนวนที่เกินจากจำนวนธนบัตรจะไม่ได้รับเงิน เช่น สถานที่ A มีธนบัตร 50,000 และ 30,000 อยู่และเมื่อจบรอบนั้น มีผู้เล่นวางลูกเต๋า 3 คน ได้แก่ นาย ก วาง 5 ลูก นาย ข วาง 1 ลูก นาย ค วาง 3 ลูก นาย ก จะได้รับ ธนบัตร 50,000 $ และ นาย ค จะได้รับธนบัตร 30,000 $ ส่วน นาย ข จะไม่ได้รับธนบัตรใบใดเลยเนื่องจากไม่มีธนบัตรเหลือแล้วในสถานที่ A
2. หากมีการวางลูกเต๋าเท่ากันเกิดขึ้นผู้เล่นทุกคนที่วางจำนวนลูกเต๋าเท่ากันทั้งหมดจะไม่ได้รับการพิจารณาในการรับเงิน

เช่น สถานที่ A มีธนบัตร 50,000 และ 30,000 อยู่และเมื่อจบรอบนั้น มีผู้เล่นวางลูกเต๋า 3 คน ได้แก่ นาย ก วาง 5 ลูก นาย ข วาง 1 ลูก นาย ค วาง 5 ลูก นาย ก และ นาย ค จะโดนตัดสิทธิ์การรับเงิน และนาย ข จะได้รับ

ธนบัตร 50,000 $ไปแทน

1. สถานที่พิเศษ(Special Location) จะไม่ได้เหมือนการพิจารณาษสถานที่ธรรมดาคือ ผู้เล่นจะได้รับเงินเท่ากับเท่ากับ 2 เท่าของธนบัตรที่ได้รับและผู้เล่นที่ได้อันดับที่1 ในสถานที่พิเศษจะสามารถเปิดการ์ดพิเศษได้ 1 ใบ ซึ่งจะมีได้แก่

- Bonus Card ซึ่งผู้เล่นได้จะได้รับเงินเพิ่ม 10,000 $

- Steal Card ผู้เล่นจะสามารถขโมยเงินผู้เล่นคนที่มีเงินมากที่สุดได้ 20,000 $ ถ้าหากคนที่ใช้การ์ดเป็นคนที่เงินมากที่สุด

จะทำการขโมยเงินผู้เล่นที่มีเงินสูงเป็นอับดับ 2 แทน

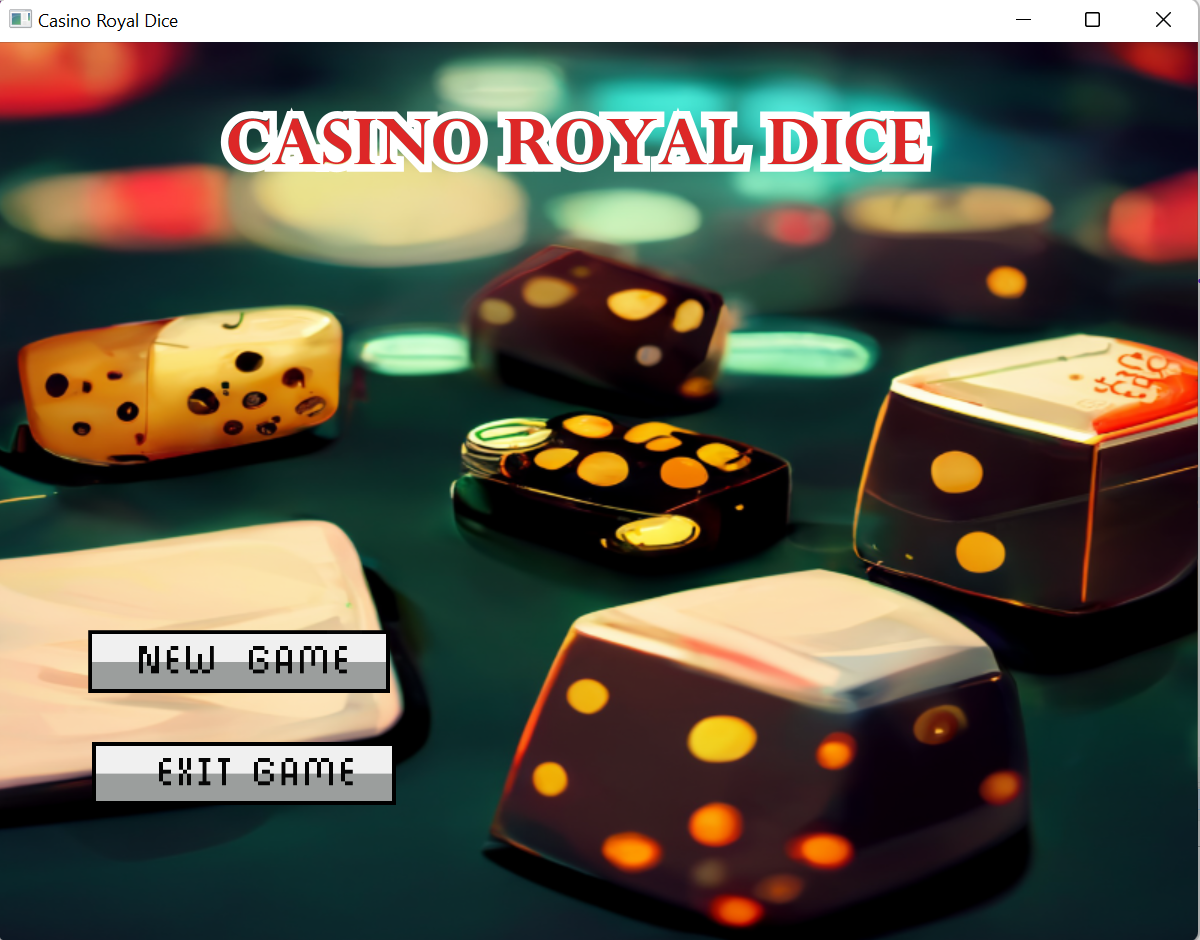
- Tax Card ผู้เล่นจะต้องเสียเงินจ่ายภาษี 10% ของเงินที่ผู้เล่นมี

- None Card การ์ดที่ไม่ได้อะไรเลยและไม่เสียอะไรเลยเช่นกัน

เมื่อพิจารณา หมดทุกสถานที่แล้วจะเริ่มรอบใหม่โดยการแจกลูกเต๋าคืนผู้เล่นทั้งหมด(8 ลูก) และใส่เงินในแต่ละสถานที่ใหม่ทั้งหมดจนกว่าแต่ละสถานที่มีเงินมากกว่า 50,000 $ เล่นตามกฎที่กล่าวมาทั้งหมด 4 ครั้ง เมื่อจบครั้งที่ 4 จะถือว่าจบเกม ผู้ชนะคือ ผู้เล่นที่มีเงินมากที่สุด

**Example**

**MainMenu scene**

****

- เมื่อกด New Game เกมจะไปหน้าถัดเลือกจำนวน Player และ ใส่ชื่อ Player

- ถ้าหากกด Exit game โปรแกรมเกมจะปิดลง

เมื่อเข้ามาถึงจะต้องเลือกจำนวนผู้เล่นก่อนและกรอกชื่อผู้เล่น ถ้าหาก กรอกชื่อน้อยกว่าจำนวนที่เลือก ผู้ที่เล่นไม่ได้ตั้งชื่อจะถูกตั้งชื่อเป็น “Player” อัตโนมัติ ถ้าหากตั้งชื่อเกินจำนวนที่เลือกชื่อที่เกินจะไม่อยู่ในเกมเมื่อเริ่มเกม

**Class diagram**

1. **Package base**
   1. Class Banknote

1.1.1 Fields

|  |  |
| --- | --- |
| **- int** banknoteValue | มูลค่าของธนบัตร |
| - **int** amount | จำนวนธนบัตร |

1.1.2 Constructor

|  |  |
| --- | --- |
| + **public** Banknote(**int** banknoteValue) | -รับมูลค่าของธนบัตร  -กำหนดจำนวนธนบัตรเริ่มต้นที่ 1 ใบ โดยใช้ setter |

1.1.3 Methods

|  |  |
| --- | --- |
| + **int** getAmount() | คืนจำนวนธนบัตร |
| + **void** setAmount(**int** amount) | กำหนดค่าธนบัตร ให้มีค่ากับ amount  โดยถ้ารับจำนวนธนบัตรเข้ามามีค่าน้อยกว่า 0 จะกำหนดจำนวนธนบัตรเป็น 0 |
| + **int** getBanknoteValue() | คืนมูลค่าของธนบัตร |
| + **void** setBanknoteValue(**int** banknoteValue) | กำหนดมูลค่าของธนบัตรให้มีค่าเป็น banknoteValue ที่รับเข้ามา |
| + **boolean** equals(Banknote other) | เปรียบเทียบ**มูลค่าธนบัตร**ว่าเท่ากันหรือไม่ |

* 1. Class CasinoBudget

1.2.1 Fields

|  |  |
| --- | --- |
| - ArrayList<Banknote> banknoteTypeList | -ArrayList เก็บ Banknote |

1.2.2 Constructor

|  |  |
| --- | --- |
| + CasinoBudget() | - สร้าง ArrayList เก็บ Banknote มูลค่า 10,000 20,000 30,000 40,000 50,000 60,000 70,000 และ 80,000 $ อย่างละ 1 ใบ |

1.2.3 Methods

|  |  |
| --- | --- |
| + ArrayList<Banknote> getBanknoteTypeList() | -คืน ArrayList ที่เก็บ Banknote โดยก่อนคืนArrayList จะทำการสลับลำดับของ Banknote  แบบสุ่ม |

* 1. Class Dice

1.3.1 Fields

|  |  |
| --- | --- |
| - **String** colour | -สีของลูกเต๋า |
| - **int** point | -แต้มของลูกเต๋าที่หงายหน้าอยู่ในขณะนั้น |

1.3.2 Constructor

|  |  |
| --- | --- |
| + Dice(String colour) | -รับสีของลูกเต๋าเข้ามา -กำหนดสีของลูกเต๋าเหมือนกับ colour ที่รับเข้ามา   โดยใช้ **setter** -กำหนดหน้าลูกเต๋าที่หงายขณะนั้นให้มีค่าเป็น 1  โดยใช้ setter |

1.3.3 Method

|  |  |
| --- | --- |
| + **void** rolling() | - สุ่มหน้าลูกเต๋าระหว่างเลข 1-6 และ กำหนดหน้า  ลูกเต๋าที่หงายอยู่ขณะนั้นให้เป็นเลขที่สุ่มได้ |
| + **String** getColour() | - คืนสีของลูกเต๋า |
| + **void** setColour(String colour) | - กำหนดสีของลูกเต๋าให้เหมือน colour ที่รับเข้ามา |
| + **int** getPoint() | - คืนหน้าของลูกเต๋าที่หงายอยู่ในขณะนั้น |
| + **void** setPoint(**int** point) | - กำหนดหน้าของลูกเต๋าที่หงายอยู่ในขณะนั้นให้มีค่าเท่ากับ point ที่รับเข้ามา |

* 1. Class Location

1.4.1 Fields

|  |  |
| --- | --- |
| - ArrayList<**Banknote**> fund | - ArrayList เก็บธนบัตรทั้งหมดของ Location |
| - ArrayList<**Integer**> diceInLocation | - ArrayList เก็บจำนวนลูกเต๋าที่ผู้เล่นวางไว้ใน  Location โดยเริ่มที่ ArrayList ช่องที่ 0 แทน  จำนวนลูกเต๋าที่วางไว้บน Location ของ ผู้เล่น  คนที่ 1 จนถึงช่องสุดท้ายแทน จำนวนลูกเต๋าที่วาง  ไว้บน Location ของผู้เล่นคนสุดท้าย |
| - **String** name | - ชื่อ Location |
| - **CasinoBudget** budget | - เงินกองกลางของเกม |
| - **int** amountOfPlayer | - จำนวนผู้เล่นทั้งหมด |
| - **int** diceValue | - ค่าของลูกเต๋าประจำ Location |

1.4.2 Constructor

|  |  |
| --- | --- |
| -Location(String name,**int** amountOfPlayer, **int** diceValue) | -รับ ชื่อสถานที่,จำนวนผู้เล่น และ หน้าลูกเต๋าประจำ  Location ตามลำดับ  - กำหนดชื่อสถานที่ให้เหมือน name ที่รับเข้ามา  โดยใช้ setter  -กำหนดจำนวนผู้เล่นให้เหมือน amountOfPlayer  ที่รับเข้ามาโดยใช้ setter  -กำหนดหน้าลูกเต๋าประจำ Location ให้เหมือน  diceValue ที่รับเข้ามาโดยใช้ setter  -สร้าง ArrayList<Banknote> เพื่อใช้เก็บธนบัตร  ทั้งหมดที่ Location มี  -สร้าง ArrayList<Integer> เพื่อเก็บลูกเต๋าของผู้เล่น  แต่ละคนที่วางไว้บน Location โดยมีช่องเท่ากับ  จำนวนผู้เล่นและทุกช่องมีค่าเป็น 0  - ตรวจสอบจำนวนเงินใน Location ว่ามีค่าน้อยกว่า  50,000 หรือไม่ โดย สามารถหาผลรวมของจำนวน  เงินที่ Location มีได้ โดยเรียก  **fundValue(fund)** ถ้าน้อยกว่าให้เพิ่มธนบัตรให้  Location เพิ่มโดยการเรียกใช้ **updateFund()**  จนกว่าจะมี จำนวนเงิน มากกว่าหรือเท่ากับ  50,000 |

1.4.3 Method

|  |  |
| --- | --- |
| + **int** fundValue(ArrayList<Banknote> fund) | - คืนผลรวมของมูลค่าธนบัตรทั้งหมดที่ Location มี |
| + **void** addDice(Player p,int amount) | -รับ Player ที่ต้องการวางลูกเต๋า และ   จำนวนลูกเต๋าที่ต้องการจะวาง  - เพิ่มจำนวนลูกเต๋าที่ผู้เล่นวางบน Location โดย  ตรวจสอบสีลูกเต๋าของผู้เล่น   ถ้าลูกเต๋า**สีขาว** แสดงถึงผู้เล่นคนแรก ให้ใส่ลูกเต๋า  ลงในช่องที่ 0 ของ diceInLocation  ถ้าลูกเต๋า**สีน้ำเงิน** แสดงถึงผู้เล่นคนที่ 2 ให้ใส่  ลูกเต๋า ลงในช่องที่ 1 ของ diceInLocation  ถ้าลูกเต๋า**สีแดง** แสดงถึงผู้เล่นคนที่ 3 ให้ใส่ลูกเต๋า  ลงในช่องที่ 2 ของ diceInLocation  ถ้าลูกเต๋า**สีเขียว** แสดงถึงผู้เล่นคนที่ 4 ให้ใส่  ลูกเต๋าลงในช่องที่ 3 ของ diceInLocation  ถ้าลูกเต๋า**สีเหลือง** แสดงถึงผู้เล่นคนที่ 5 ให้ใส่  ลูกเต๋า ลงในช่องที่ 4 ของ diceInLocation |
| + **boolean** notHaveSameElement(**int** amount, **int** index) | - รับจำนวนของลูกเต๋าเข้ามา  - ตรวจสอบว่า ใน diceInLocation มีค่าซ้ำกับ  Index อื่นหรือไม่  ถ้าซ้ำให้เปลี่ยนค่าทุกช่องที่มีค่าเท่ากับ  amount เป็น 0 และคืน false  ถ้าไม่ซ้ำไม่ต้องทำอะไรและคืน true |
| + **int** sendReward(Player p) | - ตรวจสอบว่าใน Location นั้นยังมีธนบัตรเหลือ  หรือไม่  ถ้ามี ให้เงินจำนวนเท่ากับมูลค่าของธนบัตรที่มี  มูลค่า มากที่สุดแก่ Player p และนำธนบัตรใบ  นั้นออกทำการตรวจสอบจำนวนธนบัตรถ้าธนบัตร  มูลค่า ใดมีจำนวนเป็น 0 ให้ทำการลบออกจาก  ArrayListและทำการเรียง ธนบัตรจากมูลค่ามาก  ไปน้อย  ถ้าไม่มีธนบัตรเหลือไม่ต้องทำอะไร |
| + **void** updateFund() | - จะถูกเรียกในกรณีที่ผลรวมมูลค่าธนบัตรทั้งหมดมี  ค่าน้อยกว่า 50,000 $ จะทำการเพิ่ม  ธนบัตรใบแรกสุดให้ใน budget ให้กับ Location  ก่อนจะทำการเพิ่มจะเช็คว่ามีธนบัตรมูลค่าเท่ากับ  ใบแรกสุดใน budget ใน fund หรือไม่   ถ้ามีให้ทำการเพิ่มจำนวนของธนบัตรมูลค่านั้น  ถ้าไม่มีให้ทำการสร้าง Banknote ที่มีมูลค่าเท่ากับ  ธนบัตรใบแรกสุดให้ใน budget  แล้วเพิ่มเข้าไปในfund - ทำการเรียงมูลค่าธนบัตรใน fund จากมูลค่า  มากไปน้อย |
| + **int** getAmountOfPlayer() | - คืนค่าจำนวนผู้เล่น |
| + **void** setAmountOfPlayer(**int** amountOfPlayer) | - กำหนดจำนวนผู้เล่นให้มีค่าเหมือน  amountOfPlayer  ถ้าamountOfPlayer น้อยกว่า 2 ให้กำหนดเป็น 2  ถ้ามากกว่า 5 ให้กำหนดเป็น 5 |
| + ArrayList<Banknote> getFund() | - คืน ArrayList ที่เก็บธนบัตรทั้งหมดใน Location |
| + **void** setFund(ArrayList<Banknote> fund) | - กำหนด ArrayListที่เก็บธนบัตรทั้งหมดให้  Location ให้เหมือน fund ที่รับเข้ามา |
| + ArrayList<Integer> getDiceInLocation() | - คืน ArrayList ที่เก็บลูกเต๋าของแต่ละคนที่วาง  ไว้บน Location |
| + **void** setDiceInLocation(ArrayList<Integer> diceInLocation) | - กำหนด ArrayListที่เก็บธนบัตรทั้งหมดให้  Location ให้เหมือน diceInLocation  ที่รับเข้ามา |
| + **String** getName() | - คืนชื่อของ Location |
| + **void** setName(String name) | - ตั้งชื่อของ Location ให้เหมือนกับ name ที่  รับเข้ามา ถ้ารับ Stringว่างเข้ามาให้ตั้งชื่อสถานที่  เป็น “Location” |
| + **int** getDiceValue() | - คืนค่าของลูกเต๋าประจำ Location |
| + **void** setDiceValue(**int** diceValue) | - กำหนดหน้าลูกเต๋าประจำ Location ให้มีค่า  เหมือน diceValue ที่รับเข้ามา |

* 1. Class SpecialLocation extends Location

1.5.1 Constructor

|  |  |
| --- | --- |
| +SpecialLocation(String name ,  **int** amountOfPlayer,**int** dicevalue) | -รับ ชื่อสถานที่,จำนวนผู้เล่น และ หน้าลูกเต๋าประจำ  Location ตามลำดับ  - กำหนดชื่อสถานที่ให้เหมือน name ที่รับเข้ามา  โดยใช้ setter  -กำหนดจำนวนผู้เล่นให้เหมือน amountOfPlayer  ที่รับเข้ามาโดยใช้ setter  -กำหนดหน้าลูกเต๋าประจำ Location ให้เหมือน  diceValue ที่รับเข้ามาโดยใช้ setter  -สร้าง ArrayList<Banknote> เพื่อใช้เก็บธนบัตร  ทั้งหมดที่ Location มี  -สร้าง ArrayList<Integer> เพื่อเก็บลูกเต๋าของผู้เล่น  แต่ละคนที่วางไว้บน Location โดยมีช่องเท่ากับ  จำนวนผู้เล่นและทุกช่องมีค่าเป็น 0  - ตรวจสอบจำนวนเงินใน Location ว่ามีค่าน้อยกว่า  50,000 หรือไม่ โดย สามารถหาผลรวมของจำนวน  เงินที่ Location มีได้ โดยเรียก  **fundValue(fund)** ถ้าน้อยกว่าให้เพิ่มธนบัตรให้  Location เพิ่มโดยการเรียกใช้ **updateFund()**  จนกว่าจะมี จำนวนเงิน มากกว่าหรือเท่ากับ 50,000 |

1.5.2 Method

|  |  |
| --- | --- |
| +**int** sendReward(Player p) | - ให้เงินจำนวนเท่ากับ2เท่าของมูลค่าของธนบัตรที่มี  มูลค่า มากที่สุดแก่ Player p และนำธนบัตร  ใบนั้นออก  ทำการตรวจสอบจำนวนธนบัตรถ้าธนบัตรมูลค่า  ใดมีจำนวนเป็น 0 ให้ทำการลบออกจากArrayList  และทำการเรียง ธนบัตรจากมูลค่ามากไปน้อย |

1. **Package basecard**
   1. Abstract Class Card

2.1.1 Fields

|  |  |
| --- | --- |
| - **String** name | - ชื่อการ์ด |
| - **String** keyword | - คำอธิบายความสามารถของการ์ด |

2.1.2 Constructor

|  |  |
| --- | --- |
| + Card(String name, String keyword) | - ตั้งชื่อการ์ดให้เหมือน name ที่รับเข้ามาโดยใช้  **setter**  - กำหนดคำอธิบายความสามารถการ์ดให้เหมือนกับ  keyword ที่รับเข้ามาโดยใช้ **setter** |

2.1.2 Method

|  |  |
| --- | --- |
| + **String** description() | - เรียกใช้โดย subclass |
| + **String** getName() | - คืนชื่อการ์ด |
| + **void** setName(String name) | - กำหนดชื่อการ์ดให้เหมือนกับ name ที่รับเข้ามา |
| + **String** getKeyword() | - คืนคำอธิบายความสามารถของการ์ด |
| + **void** setKeyword(String keyword) | - กำหนดคำอธิบายความสามารถการ์ดให้เหมือน  keyword ที่รับเข้ามา |

* 1. Interface Givable

2.2.1 Methods

|  |  |
| --- | --- |
| + **void** give(Player p); | - เพิ่มเงินให้กับผู้เล่น p ที่รับเข้ามา |

* 1. Interface Stealable

2.3.1 Methods

|  |  |
| --- | --- |
| + **void** steal(Player p); | - ลดเงินของผู้เล่น p ที่รับเข้ามา |

1. **Package card**
   1. class BonusCard extends Card implements Givable

3.1.2 Constructor

|  |  |
| --- | --- |
| + BonusCard(String name, String keyword) | - ตั้งชื่อการ์ดให้เป็น "Bonus Card"  - กำหนดคำอธิบายความสามารถการ์ดให้เป็น  "Congratulations!!, you got 10,000$ for free."  โดยเรียก **Constructor ของ SuperClass** |

3.1.2 Methods

|  |  |
| --- | --- |
| + **void** give(Player p) | - เพิ่มเงินให้กับผู้เล่น p ที่รับเข้ามา 10,000 $ |
| + **String** description() { | - แสดงข้อความอธิบายความสามารถการ์ดในรูปแบบ  "It's The " + **getName()** + "!! " + **getKeyword()** |

* 1. Class StealCard extends Card implements Stealable, Givable

3.2.1 Fields

|  |  |
| --- | --- |
| - int stolenMoney | - จำนวนเงินที่ต้องการหักออกจากเงินของผู้เล่น |

3.2.2 Constructor

|  |  |
| --- | --- |
| + StealCard(String name, String keyword) | - ตั้งชื่อการ์ดให้เป็น "Stealing Card"  - กำหนดคำอธิบายความสามารถการ์ดให้เป็น  "Steal $20,000 from player's money to  another player."  โดยเรียก **Constructor ของ SuperClass**  - กำหนดค่า stolenMoney ให้มีค่าเป็น  20,000 $ โดย setter |

3.2.3 Methods

|  |  |
| --- | --- |
| + **void** steal(Player p) | - หักเงินจากเงินของ Player p เท่ากับ  stolenMoney |
| + **void** give(Player p) | - เพิ่มเงินให้กับ Player p เท่ากับ  stolenMoney |
| + **int** getStolenMoney() | - คืนค่า stolenMoney |
| + **void** setStolenMoney(int stolenMoney) | - กำหนดค่า stolenMoney ให้เหมือนกับ  stolenMoney ที่รับเข้ามา |
| + **String** description() | - แสดงข้อความอธิบายความสามารถการ์ดในรูปแบบ  "OMG!! It's the " + **getName()** + ". " +  **getKeyword()** |

* 1. Class TaxCard **extends** Card **implements** Stealable

3.3.1 Constructor

|  |  |
| --- | --- |
| + TaxCard(String name, String keyword) | - ตั้งชื่อการ์ดให้เป็น "Tax Card"  - กำหนดคำอธิบายความสามารถการ์ดให้เป็น  "Meet The Revenue Department!!  Must pay 10% tax."  โดยเรียก **Constructor ของ SuperClass** |

3.1.2 Methods

|  |  |
| --- | --- |
| + **void** steal(Player p) | - หักเงิน 10% ของเงินที่ผู้เล่นมี |
| + **String** description() | - แสดงข้อความอธิบายความสามารถการ์ดในรูปแบบ  "Bad luck!! It's the " + **getName()** + ". " + **getKeyword()** |

3.4 Class NoneCard **extends** Card

1. **Package comparator**
   1. class DiceComparator implements Comparator<Dice>

4.1.1 Methods

|  |  |
| --- | --- |
| + **int** compare(Dice d1, Dice d2) | - คืนผลลบของการเปรียบเทียบหน้าของลูกเต๋า  (**Dice.getpoint()**)  - ถ้าหากหน้าลูกเต๋า d2 มากกว่า d1 จะคืนค่าจำนวน  บวก  - ถ้าหากหน้าลูกเต๋า d2 เท่ากับ d1 จะคืนค่า 0  - ถ้าหากหน้าลูกเต๋า d2 น้อยกว่า d1 จะคืนค่าจำนวน  ลบ  (ใช้ในการ Sort ArrayList<Dice> ใน class Player  จากน้อยไปมาก) |

* 1. class SortByBanknoteValue implements Comparator<Banknote>

4.2.1 Methods

|  |  |
| --- | --- |
| **+ int** compare(Banknote b1, Banknote b2) | - คืนผลลบของการเปรียบเทียบหน้าของลูกเต๋า  (Banknote.getBanknoteValue())  - ถ้าหากหน้าลูกเต๋า d2 มากกว่า d1 จะคืนค่าจำนวน  บวก  - ถ้าหากหน้าลูกเต๋า d2 เท่ากับ d1 จะคืนค่า 0  - ถ้าหากหน้าลูกเต๋า d2 น้อยกว่า d1 จะคืนค่าจำนวน  ลบ  (ใช้ในการ Sort ArrayList<Banknote> ใน class  Player จากมากไปน้อย) |

1. **Package player**
   1. class Player

5.1.1 Fields

|  |  |
| --- | --- |
| - **String** name | - ชื่อผู้เล่น |
| - **String** playerColour | - สีลูกเต๋าของPlayer |
| - **int** balance | - เก็บจำนวนเงินของผู้เล่น |
| - ArrayList<Dice> diceInPlayer | - เก็บลูกเต๋าของผู้เล่น |

5.1.2 Constructor

|  |  |
| --- | --- |
| + Player(String name, String colour) | - รับชื่อผู้เล่นและสีของลูกเต๋า  - ตั้งชื่อผู้เล่นให้เหมือน name ที่รับเข้ามาโดยใช้ **setter**  **-** กำหนดเงินเริ่มต้นให้เป็น 0 โดย **setter**  **-** กำหนดสีของลูกเต๋าให้เหมือนกับ colour ที่รับเข้า  มาโดยใช้ **setter**  **-** สร้างArrayList<Dice> เก็บลูกเต๋าที่มีสี เหมือนกับ  playerColour 8 ลูก |

5.1.3 Methods

|  |  |
| --- | --- |
| + **void** rollDice() | - หมุนหน้าลูกเต๋าแบบสุ่มให้มีค่าในช่วง 1-6 |
| + **void** sortDiceInPlayer() | - เรียงลูกเต๋าจากหน้าที่มีค่าน้อยไปหน้าที่มีค่ามาก |
| + **int** dropDice(**int** point) | - คืนจำนวนลูกเต๋าของผู้เล่นที่มีหน้าลูกเต๋าเท่ากับ  point ที่รับเข้ามา และทำการลบลูกเต๋านั้นออก  จากArrayList |
| + **void** useCardWithOutObj(Card c) | - สั่งใช้งาน BonusCard หรือ TaxCard |
| + **void** useCardWithObj(Card c,Player obj) | - สั่งใช้งาน StealCard |
| + **String** getName() | - คืนชื่อผู้เล่น |
| + **void** setName(String name) | - ตั้งชื่อผู้เล่น ให้เหมือนกับ name ที่รับเข้ามา ถ้า  หาก name รับ String ว่าง("") จะตั้งชื่อผู้เล่น  เป็น “Player” |
| + **int** getBalance() | - คืนจำนวนเงินของผู้เล่น |
| + **void** setBalance(**int** balance) | - กำหนดจำนวนเงินของผู้เล่นให้มีค่าเท่ากับ  balance ที่รับเช้ามา หาก balance ที่รับเข้ามา  น้อยกว่า 0 คืนกำหนดให้มีค่าเป็น 0 |
| + ArrayList<Dice> getDiceInPlayer() | - คืน ArrayList ที่เก็บลูกเต๋าของผู้เล่น |
| + **void** setDiceInPlayer(ArrayList<Dice> diceInPlayer) | - กำหนด ArrayList ที่เก็บลูกเต๋าของผู้เล่นให้เหมือนกับ diceInPlayer ที่รับเข้ามา |
| + **String** getPlayerColour() | - คืนสีลูกเต๋าของผู้เล่น |
| + **void** setPlayerColour(String playerColour) | - กำหนดสีลูกเต๋าของผู้เล่นให้เหมือน  playerColour ที่รับเข้ามา |

1. **Package logic** 
   1. GameLogic

6.1.1 Fields

|  |  |
| --- | --- |
| - ArrayList<Location> locationList | - ArrayList เก็บ Location |
| - ArrayList<String> locationNameList | - ArrayList เก็บชื่อ สถานที่ |
| - ArrayList<Player> playerList | - ArrayList เก็บผู้เล่นทุกคน |
| - **int** roundCount | - ลำดับรอบที่กำลังเล่นอยู๋ |
| - **int** indexPlayer | - index ที่แทนลำดับของผู้เล่น |
| - **boolean** isRoll | - |
| - **Boolean** selected | - |
| - **Boolean** cardSeal | - เก็บค่าว่ากองการ์ดอนุญาตให้เปิดหรือไม่ |
| - **Boolean** waiting | - |
| - Player currentPlayer | - |
| - Player playerWinSpecial | - ผู้เล่นที่ได้อันดับที่ 1 ใน Special Location |
| - **int** curretntDiceSelect | - |
| - **int** amountOfPlayer | - จำนวนผู้เล่น |
| - CardDeck cardDeck | - กองการ์ดสำหรับผู้เล่นที่ได้อันดับที่ 1 ใน  Special Location |
| - ArrayList<Integer> oldBalanceList | - |

6.1.3 Methods

|  |  |
| --- | --- |
| + **void** updateGameStatus(String string, Color color) | - แสดงข้อความบอกStatusของเกมว่าผู้เล่นทำอะไร  และตัวเกมต้องการให้ผู้เล่นทำอะไร  (กล่องข้อความด้านขวาล่าง) |
| + **void** initialScoreBoard() | - สร้างกระดานคะแนน ที่แสดงชื่อผู้เล่น จำนวนเงินที่  แต่ละคนมีและจำนวนลูกเต๋าที่อยู่กับผู้เล่น |
| + **void** updateScoreBoard() | - update ScoreBoard เมื่อเกิดการเปลี่ยนแปลง |
| + **void** updateBalanceStatus() | - update จำนวนเงินรวมของผู้เล่น |
| + **void** updateDice(Player player) | - update จำนวนลูกเต๋าที่ยังอยู่กับผู้เล่น |
| + **void** roll() |  |
| + **void** newGame(**int** amount) |  |
| + **void** playGame(ArrayList<Player> playerList) |  |
| + **void** endRound() | - จะถูกเรียกต่อเมื่อลูกเต๋าของผู้เล่นหมดทุกคนและยัง  ไม่ใช่รอบที่ 4 จะทำแจกเงินจากแต่ละ Location ให้  ผู้เล่น โดยผู้เล่นคนที่มีลูกเต๋าบน Location นั้นๆ จะ  ได้รับธนบัตรที่มีมูลค่ามากที่สุดที่ Location นั้นมี  และพิจรณาผู้เล่นอื่นกับธนบัตรที่เหลือตามลำดับ  จากจำนวนลูกเต๋าที่วางอยู่ใน Location นั้น  - ถ้าหากเป็น SpecialLocation จะทำเหมือน ทุก  อย่างเหมือนที่กล่าวไปด้านบนแต่จะมีการ ?เพิ่มไพ่?  - สุดท้ายจะทำการเรียก **Gamelogic.resetBoard**  เพื่อเตรียมสำหรับเกมรอบถัดไป และเรียก  **Gamelogic.playgame** เพื่อเริ่มเกมรอบถัดไป |
| + **void** resetBoard(**int** amountOfPlayer) | - ทำการ reset ทุกอย่างบนBoard ให้เป็น  เหมือนก่อนเริ่มเกม,สุ่ม SpecialLocation ใหม่  และทำการ update คะแนนของผู้เล่นใน  รอบที่ผ่านมา |
| + **void** checkGameEnd() | - |
| + **void** clickNextRound(MouseEvent event) | - |
| + **void** addBankNoteToLocation() | - เพิ่มธนบัตรให้กับแต่ละ Location ทั้งหมด  จนกว่าแต่ละ Location จะมีมูลค่าธนบัตร  มากกว่า 50,000 $ |
| + **void** resetCard() | - reset กองการ์ดให้เป็นเหมือนก่อนเริ่มเกม |
| + **void** openCard(MouseEvent event) | - ทำการเปิดการ์ดใบบนสุดโดยจะเปิดได้ก็ต่อเมื่อ  เป็นผู้เล่นที่ได้อันดับที่ 1 ใน Special Location  ในรอบนั้นๆ  ถ้าไม่อนุญาตให้แล้วกดเปิดจะขึ้นข้อความ  "You can't open card at this time." |
| + **void** cardEffect(Card card) | - สั่งใช้งานความสามารถการ์ดทุกประเภท ยกเว้น  None Card |
| + **void** setNewSpecialLocation() | - สุ่มกำหนดว่าจะให้ Location ใดให้เป็น  Special Location |
| + **void** openCard(MouseEvent event) | - จะถูกเรียกต่อเมื่อผู้เล่นทำการเปิดการ์ด |
| + **void** initialLocationList(**int** amount) | - สร้าง Location จำนวน 6 ที่และใส่ใน ArrayList  และเก็บไว้ใน ArrayList<Location> locationList |
| + **void** setNewLocationList(**int** locationCode) | - จะถูกเรียกใน  **GameLogic.setNewSpecialLocation()**  เพื่อสร้าง SpecialLocation แทนที่ Location ใน  ArrayList<Location> locationList ที่ตำแหน่ง  indexที่สุ่มได้ |
| + **void** resetLocation() | - |
| + **void** resetDiceImgInLocation() | - ทำการเอารูปลูกเต๋าของผู้เล่นทุกคนออกจาก  Location |
| + **void** resetAmountOfDiceInLocation() | - |
| + **void** resetPlayerDice() | - ทำให้ลูกเต๋าที่ผู้เล่นมีกลับเป็น 8 ลูกทุกคน |
| + **void** clearBankNoteInLocation() | - ทำให้ ArrayList<Banknote> ใน Location แต่ละที่กลายเป็น ArrayList ว่าง |
| + **void** selectLocation(MouseEvent event) | - |
| + **void** dropDiceInLocation(**int** numberLocation, Player player, ArrayList<Text> textList) | - แสดงข้อความว่าผู้เล่นคนใดว่างลูกเต๋ากี่ลูกในสถานที่ใดในรูปแบบดังนี้ -กรณี 1 ลูก  Playername+ " drop a dice in " + Locationname  -กรณีมากกว่า 1 ลูก  Playername+ " drop" +amount+ " dices in "+ Locationname  และ...??? |
| + **void** selectDice(MouseEvent event) | - จะถูกเรียกใช้เพื่อทำให้ลูกเต๋าเลขหน้าอื่นๆที่ไม่ใช่  เลขหน้าลูกเต๋าที่ผู้เล่นกดเป็นลูกเต๋าเป็นสีเทา |
| + Player findRichestPlayerWithOut(Player player) | - คืนผู้เล่นที่มีจำนวนเงินมากที่สุด |
| + **void** editSelectDice(**int** dicePoint, Player player) | - |
| + **boolean** allOutOfDice() | - จะคืน true ต่อเมื่อ ผู้เล่นทุกคนไม่มีลูกเต๋าเหลือ  ที่ผู้เล่นแล้ว |
| + ArrayList<String> getLocationNameList() | - คืน ArrayListที่เก็บชื่อสถานที่ |
| + **void** setLocationNameList(ArrayList<String> locationNameList) | - กำหนด ArrayListเก็บชื่อสถานที่ ให้เหมือนกับ  locationNameList ที่รับเข้ามา |
| + **int** getRoundCount() | - คืนจำนวนรอบการเล่นที่เล่นจบไปแล้ว |
| + **void** setRoundCount(**int** roundCount) | - กำหนดจำนวนรอบที่เล่นจบไปแล้วให้เท่ากับ  roundCount ที่รับเข้ามา |