UCLA

CS 31 : Introduction To Computer Science I

Howard A. Stahl

1

Project 6

- The Goal: A Working Slot Machine Game
- Background: Please Play A Few Games With This Free Game
  - http://www.freeslots.com
- Truth In Advertising:
  - We'll Only Be Dealing With The Following Concepts: RandomNumber, Screen, Bank, PayTable, SlotMachine
  - No Need To Worry Sound, Graphics, Bonus Round

2

Project 6

- Unlike Earlier Assignments, I Am Supplying You With A Partial "Skeleton" Of The Code Solution
- It Will Run Right Out Of The Box
  - Some Important Pieces Are Stubbed Out…
  - These Are The Parts You Need To Complete
- Hint 1: Acquire The Skeleton!
- Hint 2: Build And The Run The Skeleton!
  - Look At What Is Working And What Is Not
- Hint 3: Use CodeBoard Once You Have It All Running!
  - You Can Paste All Your Class Code Into It To Run Some Tests I Made For You

3

Project 6

- The Work Product: The Implementation Of The Public API Of The Classes Described Here And In The Assignment.
- You Are Free To Do It However You Like, But You Must Provide The Public API I Am Looking For...
  - You Can Add Classes, Methods, Members As You Feel Appropriate
  - But I Honestly Don't Think You'll Need To…
- In What Follows, It Is The **Bolded** Portions That You Need To Complete

4

The RandomNumber Class

- Using The RandomNumber Class, We'll Have Random Behavior, Like In The Real World…

RandomNumber
-mMinimum : int -mMaximum : int
+RandomNumber(min : int, max : int, minInclusive : bool = true, maxInclusive : bool = true) +random( ) : int

5

The RandomNumber Class

- Using The RandomNumber Class, We'll Have Random Behavior, Like In The Real World…

RandomNumber
-mMinimum : int -mMaximum : int
+RandomNumber(min : int, max : int, minInclusive : bool = true, maxInclusive : bool = true) +random( ) : int

6

The Screen Class

- Using The Screen Class, We'll Display Spinning Wheels And Wager Information As Play Proceeds…

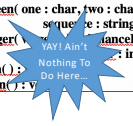
Screen
+ Screen( ) + displayScreen(one : char, two : char, three : char, sequence : string) + displayWager(wager : int, balanceBefore : int, balanceAfter : int) + clearScreen( ) : void + pauseScreen( ) : void

7

The Screen Class

- Using The Screen Class, We'll Display Spinning Wheels And Wager Information As Play Proceeds…

Screen
+ Screen( ) + displayScreen(one : char, two : char, three : char, sequence : string) + displayWager(wager : int, balanceBefore : int, balanceAfter : int) + clearScreen( ) : void + pauseScreen( ) : void



8

The Bank Class

Bank
- bankAmount : int - wager : int
+ Bank(amount : int = 0) + win(amount : int) : void + lose(amount : int) : void + deposit(amount : int) : void + balance( ) const : int + cashOut( ) : int + canWager(amount : int) const : bool + setWager(amount : int) : void + getWager( ) const : int

- Manages Credits And Wagers
- This Class Is Stubbed Out…
  - So Code Builds But You Need To Complete It
- `bankAmount` Are Game Tokens
  - Added To The Bank With Calls To **`deposit(int)`**
  - Pulled From The Bank With Calls To **`cashOut( ) : int`**
  - Accessor Method: `balance( ) : int`
  - The Bank Constructor Allows For Someone To Start With "Free" credits
- `wager` Are Game Tokens Bet On The Next Play Of The Machine
  - Accessor Method: `getWager( ) : int`
  - Mutator Method: `setWager( int )`
  - Validation Method: **`canWager( int ) : bool`**

9

The Bank Class

Bank
- bankAmount : int - wager : int
+ Bank(amount : int = 0) + win(amount : int) : void + lose(amount : int) : void + deposit(amount : int) : void + balance( ) const : int + cashOut( ) : int + canWager(amount : int) const : bool + setWager(amount : int) : void + getWager( ) const : int

- Manages Credits And Wagers
- This Class Is Stubbed Out…
  - So Code Builds But You Need To Complete It
- `bankAmount` Are Game Tokens
  - Added To The Bank With Calls To **`deposit(int)`**
  - Pulled From The Bank With Calls To **`cashOut( ) : int`**
  - Accessor Method: `balance( ) : int`
  - The Bank Constructor Allows For Someone To Start With "Free" Credits
- `wager` Are Game Tokens Bet On The Next Play Of The Machine
  - Accessor Method: `getWager( )`
  - Mutator Method: `setWager( int )`
  - Validation Method: **`canWager( int ) :`**

Whole Dollar-Based Bank
No Coins…

10

The Bank Class

Bank
- bankAmount : int - wager : int
+ Bank(amount : int = 0) + win(amount : int) : void + lose(amount : int) : void + deposit(amount : int) : void + balance( ) const : int + cashOut( ) : int + canWager(amount : int) const : bool + setWager(amount : int) : void + getWager( ) const : int

- Manages Credits And Wagers
- `wager` Are Game Tokens Bet On The Next Play Of The Machine
  - Validation Method: **`canWager( int )`**
    - Return `false` If Someone Tries To Bet More Than What They Have In The Bank…
    - Return `true` Otherwise…
- Eventually, A Round Of Play Will Occur And The Slot Machine Will Tell The Bank What Happened Via Calls To…
  - **`win( int )`** -> The Player Won! So Increase The `bankAmount` …
  - **`lose( int )`** -> The Player Lost! So Decrease The `bankAmount` …

11

The Bank Class

Bank
- bankAmount : int - wager : int
+ Bank(amount : int = 0) + win(amount : int) : void + lose(amount : int) : void + deposit(amount : int) : void + balance( ) const : int + cashOut( ) : int + canWager(amount : int) const : bool + setWager(amount : int) : void + getWager( ) const : int

- You Need To Get This Class Working First…
- The Assignment Provides Sample Driver Code And `asserts( )` To Be Sure Things Are Working Right…
- CodeBoard Provides Tests As Well...

12

The PayTable Class

PayTable
- mWheel1 : char - mWheel2 : char - mWheel3 : char
+ PayTable(wheel1 : char, wheel2 : char, wheel3 : char) + calculateMultiplier( ) : Multiplier + manageWager(bank : Bank &) : void

- Manages Payouts
- This Class Is Stubbed Out…
  - So Code Builds But You Need To Complete It
- You Provide The Three Wheel Values At Constructor-Time
- **`calculateMultiplier( )`** Determines The Pay Out By Returning A Multiplier Value
- **`manageWager( Bank & )`** Updates A Bank, Based On The Wager, The Balance And The Multiplier

13

Introducing The Multiplier Enumeration

- The Multiplier Represents A Pay Line On The Machine…
- It Is `PayTable::Multiplier` And Should Be `public`

<<enumeration>> Multiplier
ZERO, ONETIME, TWOTIME, THREETIME, FOURTIME, FIVETIME, SIXTIME, SEVENTIME, TENTIME

14

Introducing The Multiplier Enumeration

- The Multiplier Represents A Pay Line On The Machine…
- It Is `PayTable::Multiplier` And Should Be `public`

<<enumeration>> Multiplier
ZERO, ONETIME, TWOTIME, THREETIME, FOURTIME, FIVETIME, SIXTIME, SEVENTIME, TENTIME

Of Course, A Losing Round Should Have A Multiplier Of `PayTable::Multiplier::ZERO`…

15

Introducing The Multiplier Enumeration

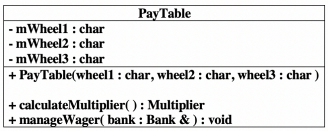
- Table Of Pays (Winning Plays…)

Description	Winning Rate	`calculateMultiplier( )` returns
a single Ace	1-to-1	`Multiplier::ONETIME`
two Aces	5-to-1	`Multiplier::FIVETIME`
any pair other than Aces	3-to-1	`Multiplier::THREETIME`
any pair other than Aces with an Ace	4-to-1	`Multiplier::FOURTIME`
three Aces	10-to-1	`Multiplier::TENTIME`
three of a kind other than Aces	7-to-1	`Multiplier::SEVENTIME`
Royal Straight:AKQ in any order	5-to-1	`Multiplier::FIVETIME`
Anything Else	None	`Multiplier::ZERO`

16

The PayTable Class

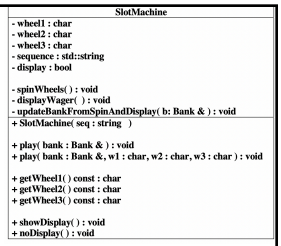
- You Need To Get This Class Working Next…
- The Assignment Provides Sample Driver Code And `asserts( )` To Be Sure Things Are Working Right…
- CodeBoard Provides Tests As Well...



17

The SlotMachine Class

- Our SlotMachines Have 3 Wheels…
- Our SlotMachines Use A Bank To Play…
- Our SlotMachines Have A Sequence Of Letters That Get Displayed To Show The Wheels Spinning…
- For Interactive Play, `showDisplay()` Will Use The Class Screen To Print The Wheels. For Silent Play, `noDisplay()` Generates No Output Which Is Good For Testing…



18

The SlotMachine Class

- Our SlotMachines Have 3 Wheels…
- Our SlotMachines Use A Bank To Play…
- Our SlotMachines Have A Sequence Of Letters That Get Displayed To Show The Wheels Spinning…
- For Interactive Play, `showDisplay()` Will Use The Class Screen To Print The Wheels. For Silent Play, `noDisplay()` Generates No Output Which Is Good For Testing…

SlotMachine

- wheel1 : char
- wheel2 : char
- wheel3 : char
- sequence : std::string
- display : bool

- spinWheels( ) : void
- displayWager( ) : void
- updateBankFromSpinAndDisplay(b: Bank &) : void

+ SlotMachine(seq : string)

+ play(bank : Bank &) : void
+ play(bank : Bank &, w1 : char, w2 : char, w3 : char) : void

+ getWheel1( ) const : char
+ getWheel2( ) const : char
+ getWheel3( ) const : char

+ showDisplay( ) : void
+ noDisplay( ) : void

19

The SlotMachine Class

- Our SlotMachines Have 3 Wheels…
- Our SlotMachines Use A Bank To Play…
- Our SlotMachines Have A Sequence Of Letters That Get Displayed To Show The Wheels Spinning…
- For Interactive Play, `showDisplay()` Will Use The Class Screen To Print The Wheels. For Silent Play, `noDisplay()` Generates No Output Which Is Good For Testing…

No Need To Touch `spinWheels( )` Or `displayWager( )` But Expect To Call Them…

SlotMachine

- wheel1 : char
- wheel2 : char
- wheel3 : char
- sequence : std::string
- display : bool

- spinWheels( ) : void
- displayWager( ) : void
- updateBankFromSpinAndDisplay(b: Bank &) : void

+ SlotMachine(seq : string)

+ play(bank : Bank &) : void
+ play(bank : Bank &, w1 : char, w2 : char, w3 : char) : void

+ getWheel1( ) const : char
+ getWheel2( ) const : char
+ getWheel3( ) const : char

+ showDisplay( ) : void
+ noDisplay( ) : void

20

The SlotMachine Class

- Trivial Accessors:
  - `getWheel1( ) : char`
  - `getWheel2( ) : char`
  - `getWheel3( ) : char`
- **`play(...)`** Is The Major Operation
  - Spin The Wheels, If Desired
  - Adjust The Wheels For The Round Of Play
  - Determine PayTable Multiplier
  - Update The Bank
  - Display Updated Wager Information, If Desired

SlotMachine

- wheel1 : char
- wheel2 : char
- wheel3 : char
- sequence : std::string
- display : bool

- spinWheels( ) : void
- displayWager( ) : void
- updateBankFromSpinAndDisplay(b: Bank &) : void

+ SlotMachine(seq : string)

+ play(bank : Bank &) : void
+ play(bank : Bank &, w1 : char, w2 : char, w3 : char) : void

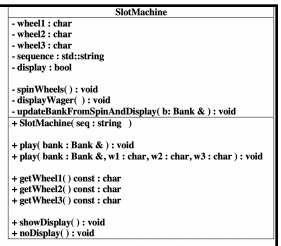
+ getWheel1( ) const : char
+ getWheel2( ) const : char
+ getWheel3( ) const : char

+ showDisplay( ) : void
+ noDisplay( ) : void

21

The SlotMachine Class

- Why Are There Two Versions Of Play?
- One Is For Cheating…



22

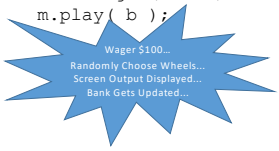
Driver Code Says:

```
SlotMachine m;
m.showDisplay( );
Bank b( 100 );
b.setWager( 100 );
m.play( b );
```

23

Driver Code Says:

```
SlotMachine m;
m.showDisplay( );
Bank b( 100 );
b.setWager( 100 );
m.play( b );
```



24

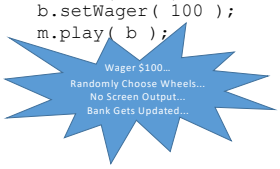
Driver Code Says:

```
SlotMachine m;
m.noDisplay( );
Bank b( 100 );
b.setWager( 100 );
m.play( b );
```

25

Driver Code Says:

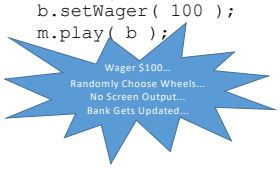
```
SlotMachine m;
m.noDisplay( );
Bank b( 100 );
b.setWager( 100 );
m.play( b );
```

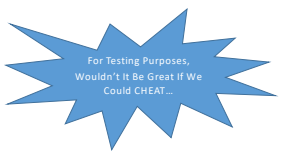


26

Driver Code Says:

```
SlotMachine m;
m.noDisplay( );
Bank b( 100 );
b.setWager( 100 );
m.play( b );
```





27

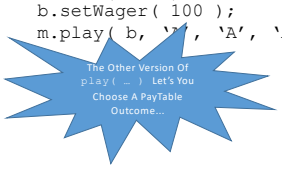
Cheating Driver Code Says:

```
SlotMachine m;
m.noDisplay( );
Bank b( 100 );
b.setWager( 100 );
m.play( b, 'A', 'A', 'A' );
```

28

Cheating Driver Code Says:

```
SlotMachine m;
m.noDisplay( );
Bank b( 100 );
b.setWager( 100 );
m.play( b, 'A', 'A', 'A' );
```



29

Cheating Driver Code Says:

```
SlotMachine m;
m.noDisplay( );
Bank b( 100 );
b.setWager( 100 );
m.play( b, 'A', 'A', 'A' );
assert( b.balance() == 1100 );
```

30

Suggestions

- In The Assignment, Scroll Down And Review The `assert( )` Commands…
- Start With Bank…
- Then Move On To PayTable…
- And Finish With SlotMachine!
- `assert( )` Each Class And Its Methods As You Make Progress…
- Don't Finally Play The Game Until All Your Classes Pass Their `assert( )`'s
- You Can Check Your Work To Some Degree Via CodeBoard As Well…

31