Trustwave

Horse Track Programming Assignment

PROBLEM DESCRIPTION:

Your task is to develop a simulation for an automated teller machine for Horse Race Park.

The teller machine you develop only has two responsibilities. The first is to allow a park employee to set the number of the winning horse. The second is to allow patrons to determine if the horse they have already bet on won, and then collect their winnings if it did.

Upon startup, the machine should display a list of its current inventory of money, followed by a list of horses and current race results. At this point the program accepts user input.

The specified input and output formats for the machine must be followed exactly. At the end of these instructions, you will find examples of some input/output scenarios.

PROCESSING REQUIREMENTS:

The calculation for ticket winnings must take the odds into account. The following table lists the available horses and their associated odds:

#	Horse Name	Odds
1	That Darn Gray Cat	5
2	Fort Utopia	10
3	Count Sheep	9
4	Ms Traitour	4
5	Real Princess	3
6	Pa Kettle	5
7	Gin Stinger	6

The machine must dispense the minimum number of bills to complete a payout, constrained by its current inventory. If the machine can't dispense a payout due to insufficient funds, an error message is displayed (see below).

The following table lists the denominations the machine holds and the starting inventory:

Denomination	Inventory
$1	10
$5	10
$10	10
$20	10
$100	10

The machine can be restocked to its initial state at any time. Restocking the machine should restore each denomination to the initial inventory amount.

There isn't any limit on how much a patron can wager on a particular race, but all bets must be in increments of $1. For example, $1, $3, $10, $1000 are all valid bets, but $1.50 isn't a valid bet. You must display an error message if the user enters an invalid bet.

INPUT FORMAT:

Your solution should read from the standard input stream, one command per line. No prompts or other extraneous user messages should be displayed. Blank input lines should be ignored.

Valid commands are as follows:

- 'R' or 'r' - restocks the cash inventory
- 'Q' or 'q' - quits the application
- 'W' or 'w' [1-7] - sets the winning horse number
- [1-7] <amount> - specifies the horse wagered on and the amount of the bet

If the user enters an improperly formatted command, then the program should display a single-line message with the following format:
`Invalid Command: <characters that were entered>`
If the user tries to specify a non-existent horse number, then the program should display a single-line message with the following format:
`Invalid Horse Number: <number that was entered>`
If the user enters a horse number that didn't win, the program should display a single-line message with the following format:
`No Payout: <horse name>`
If the user enters an invalid bet amount, the program should display a single-line message with the following format:
`Invalid Bet: <amount>`
If the user enters a horse number that did win, the program should display a multi-line message with the following format:

```
Payout: <horse name>,<total winnings>
Dispensing:
$1,<number of $1 bills>
$5,<number of $5 bills>
$10,<number of $10 bills>
```

```
$20,<number of $20 bills>
$100,<number of $100 bills>
```
However, if the machine does not have enough money to make the complete payout, the program should display a single-line message with the following format:
```
Insufficient Funds: <payout amount>
```
The machine inventory and list of horses (see next section) should be displayed immediately following any applicable message.

At application startup horse number 1 is considered the winner. The winner can be changed using the 'w' command as outlined above.

OUTPUT FORMAT:

All output should be written to the standard output stream. At program startup, and following the processing of every command, the machine inventory and the horse list should be displayed. Both the inventory and horse lists should be ordered as shown above
```
Inventory:
<denomination>,<quantity in inventory>
...
<denomination>,<quantity in inventory>
Horses:
<horse number>,<horse name>,<odds>,<did-win>
...
<horse number>,<horse name>,<odds>,<did-win>
```
The `<did-win>` indicator should be either "won" or "lost".

Note: the sample output is indented in these instructions to make it easier to read. The output generated by your program should *not* have any whitespace at the beginning of a line.

TECHNICAL NOTES:

Your solution should be a command-line program written in C++ using Microsoft Visual Studio 2015 or later. If you use any external libraries in developing your solution then you should bundle these libraries with your code so that we can run your solution.

It is *not* required that the initial machine configuration (inventory counts, list of horses, etc.) be dynamic. In particular, it is acceptable to perform this initialization in code, rather than reading the configuration from an external file or database. However, your program should be flexible enough to allow inventory or horses to be added or deleted without requiring extensive code changes.

Make sure your program works correctly for all combinations of inputs. You may include automated tests if you like (using a framework such as JUnit or NUnit), but this is not required.

Extensive inline or method-level comments are not required, unless you want to include them to highlight particular aspects of your design or implementation.

EXAMPLE:

Upon application startup, the initial inventory and horse lists would look like this:

```
Inventory:
$1,10
$5,10
$10,10
$20,10
$100,10
Horses:
1,That Darn Gray Cat,5,won
2,Fort Utopia,10,lost
3,Count Sheep,9,lost
4,Ms Traitour,4,lost
5,Real Princess,3,lost
6,Pa Kettle,5,lost
7,Gin Stinger,6,lost
```

For input consisting of the following commands:

```
1 55
2 25
W 4
4 10.25
```

the program would produce the following output (including the startup output):

```
Inventory:
$1,10
$5,10
$10,10
$20,10
$100,10
Horses:
1,That Darn Gray Cat,5,won
2,Fort Utopia,10,lost
3,Count Sheep,9,lost
4,Ms Traitour,4,lost
5,Real Princess,3,lost
6,Pa Kettle,5,lost
7,Gin Stinger,6,lost
Payout: That Darn Gray Cat,$275
Dispensing:
$1,0
$5,1
$10,1
$20,3
$100,2
Inventory:
$1,10
$5,9
$10,9
$20,7
$100,8
Horses:
1,That Darn Gray Cat,5,won
2,Fort Utopia,10,lost
3,Count Sheep,9,lost
4,Ms Traitour,4,lost
```

```
5,Real Princess,3,lost
6,Pa Kettle,5,lost
7,Gin Stinger,6,lost
No Payout: Fort Utopia
Inventory:
$1,10
$5,9
$10,9
$20,7
$100,8
Horses:
1,That Darn Gray Cat,5,won
2,Fort Utopia,10,lost
3,Count Sheep,9,lost
4,Ms Traitour,4,lost
5,Real Princess,3,lost
6,Pa Kettle,5,lost
7,Gin Stinger,6,lost
Inventory:
$1,10
$5,9
$10,9
$20,7
$100,8
Horses:
1,That Darn Gray Cat,5,lost
2,Fort Utopia,10,lost
3,Count Sheep,9,lost
4,Ms Traitour,4,won
5,Real Princess,3,lost
6,Pa Kettle,5,lost
7,Gin Stinger,6,lost
Invalid Bet: 10.25
```