**Problem Statement**

**BACKGROUND:**

You have been hired to develop a program for the newly reopened West Point Train Station. Authorities are considering a new passenger train service along the west side of the Hudson River, and they are starting with a 3-stop line connecting Newburg, West Point, and Nyack. Eventually, the line will be extended to Albany and New York City, and a highly sophisticated train sensing and ticketing system (compatible with the other Metro stations) will be put into service. However, right now the West Point Train Station needs a temporary system with a few basic features. This system will be used by the cashier, as she conducts transactions with customers and observes arrivals of trains at the West Point station.

**FUNCTIONAL REQUIREMENTS:**

The system that you will design, implement, and test will be a text-based terminal driven by a menu. The menu (displayed as soon as the program is started) must allow the cashier to select from several possible actions:

1) Sell tickets to Newburg

2) Sell tickets to Nyack

3) Process train departure

4) Check tickets remaining

5) View total sales

6) Quit this program

From the West Point station, tickets to Newburg cost $3.50 and tickets to Nyack cost $5.25. The cashier will use menu choices #1 and #2 to conduct the sales transaction. To start the sales transaction, the cashier will need to enter the number of tickets that the customer wants to purchase. The system will then display the total price for the tickets. Next, the cashier will enter the total paid by the customer and the system will specify the change due to the customer in dollars, quarters, dimes, nickels, and pennies or show a message “Exact price paid.” Once the sales transaction is complete, the program must deduct the tickets just sold from the total tickets available and then redisplay the menu.

The West Point station can sell up to 10 tickets for each Newburg train and 15 tickets for each Nyack train. Since train schedules are not yet predictable, tickets can only be sold for one train at a time going to either Newburg or Nyack. However, once the cashier has seen a train arrive and depart from the station, she must be able to use menu choice #3 in your program to note the departure of the train and start selling tickets for the next train.

The cashier will use menu choice #4 to count off each ticket left for both trains, displaying each count to the screen. For example, if the next Newburg train has 2 tickets left and the next Nyack train has 3 tickets left, the system will show:

“1 Newburg ticket

2 Newburg ticket

1 Nyack ticket

2 Nyack ticket

3 Nyack ticket”

At any point, the cashier should also be able to use menu choice #5 to display the dollar value sum of ticket sales since the start of the program. The program should only terminate when menu option #6 is selected.

**TECHNICAL REQUIREMENTS:**

Your solution will consist of deliverables in two phases:

1. Phase 1: Design (50 points, to include a Problem Specification, Flowchart, Functional Decomposition, and Partial Test Plan) **due at 1600 hrs on Lesson 14.**
2. Phase 2: Implementation (150 points, Ada code and completed Test Plan) **due at 1600 hrs on Lesson 22.**

Beyond satisfying the functional requirements, there are a few specific technical requirements that you must meet to earn full credit. Your instructor may have additional instructions as well.

* Your solution must exhibit top-down design, follow all naming conventions (see the Ada95 Style Guide, linked from the CS300 Tools page), and be documented IAW the USMA pamphlet *Documentation of Written Work (DWW).* This project is expected to be an individual project, so you must document all assistance that you receive.
* Your design and implementation must be thoroughly tested. Make sure that you include test cases for each menu choice, input validation, and path testing.
* Your implementation must use a single function to calculate how many of each denomination of currency is needed for change. There are a variety of acceptable ways to implement this function—strive to make your function be a general calculation process that reduces repetitious lines of code.
* Declare an enumerated type for the two trains, and use it to determine which train ticket counter to reset in menu choice #3.
* You must use two different types of iterative programming constructs to implement the ticket counting processes (menu choice #4).