Suppose you are working on a turn-based (i.e. user makes a move, then the CPU makes a move (much like chess) war strategy game. They all have the following attributes:

1. Range: the distance upto which it can attack enemies
2. Health Point (HP): the unit dies when HP drops to 0 or a lower value
3. Cost: the amount of coins needed to deploy it
4. Damage: the amount of damage it can inflict upon its enemies per turn

There are two type of units:

- Moving Unit
- Stationary Unit

Archers and Advanced Archers are part of the Moving unit where Bowman and Advanced Bowman are part of the Stationary unit. Bowmen shoots basic arrows and Advanced Bowmen shoots using canon. On the other hand, Archers shoot advanced arrows and Advanced Archers shoot improved arrows. Although the stationary unit cannot move, because of their high shooting range, they can be pretty handy. The Archers move by running on the field whereas the Advanced Archers ride horses. The advanced archers are much like Dothraki from Game of Thrones. They are very efficient in war and can cause great damage to the opponent; making them an excellent troop. Whenever you deploy a force, it will cost you some coin. The stationary unit has one advantage. Whenever they die, they will give you some coin back.

Besides these, there is an Enemy Tower which has tremendous defensive mechanism. The Enemy tower remains still but its shooting range is extremely high. A comprehensive summary is given in the following table:

Table 1: Description of different units.

unit	range	health	Damage	Cost	Step size	Return Coin	Shoot Type	movement
bowman	110	60	10	100	N/A	40	Basic Arrow	N/A
advanced bowman	130	85	15	200	N/A	60	Canon	N/A
archer	50	100	25	150	25	N/A	Advanced Arrow	Run
advanced archer	50	120	50	600	30	N/A	Improved Arrow	Ride Horse
Enemy Tower	200	300	40	N/A	N/A	N/A	Fire Arrow	N/A

Gameplay

The game progresses in rounds, each round consisting of two turns (the first turn for the user and the second one for the CPU). Suppose right now you are interested to test a simplified version of the game. In this version there is a single enemy tower (which is stationary) that you need to destroy. The tower is at 100 units distance from your base camp. You can generate as many units as you want (within the coin limit) one after another (i.e. after one unit dies, you can generate another one), but you cannot generate more than one unit simultaneously. That means only one of your units will be active at a time. After it is destroyed, you can generate the next one. Also remember, it will cost you some coin to generate a unit. You can choose which unit you want to generate. In each turn, a unit moves towards the enemy tower in order to get as close to it as possible, but never crosses it. It then shoots the target if it is within range. If the unit dies, user can choose the next unit to generate. Every time the newly generated unit will start from the beginning. The user has a fixed amount of coin initially which in this case is 1600. If all the coins get spent during the game, the user will lose. If the user can destroy the enemy tower, he will be the winner.

Instructions

You must use your concept of inheritance appropriately. The class hierarchy should be maintained properly. A small code fragment is given. In the code segment, you can see that we have created a base class pointer w**. You have to complete all the functionalities of the user using only the base class pointer w.** You cannot instantiate any of your derived classes. That means you cannot write the following code fragment or something similar to this:

```
Archer arch;
```

If you complete this assignment correctly, you will be able to understand the full power of inheritance. For the enemy tower, you may instantiate another base class pointer. However, you cannot instantiate any derived class for that as well. You have been given an exe of the solution. You can run the exe file to get a better understanding of what you have to do. Try to show your outputs according to the output of the exe file.

Submission Deadline: 01/12/18 (01:00 AM)

Submission Guideline: Create a zip file having only the cpp file(s) of your program. The file’s name should be your student id.