# Hints for Project 3: Competition

1. What data members are in this class?

Answer: a competition involves a hare, a tortoise, and a road. Since a hare can be created from a default or non-default constructor, we use a pointer to Hare class as a data member. Similarly, we have Tortoise\* and Road\* type as data members.

1. What are operations of Competition class?

Answer: Define start method to mimic a competition. In addition, define constructors and destructor, even though there are not function members.

1. How do we define default constructor of Competition class?

Suppose data member of Competition are

Hare\* hare;

Tortoise\* tor;

Road\* rd;

Then default constructor is defined as follows.

Competition::Competition()

{

srand(1);

hare = **new** Hare();

tor = **new** Tortoise();

rd = **new** Road();

}

1. How do we define non-default constructor of Competition?

Warning: since we use autograder, it is hard to test codes with random result, so we fix a random sequence by using srand(1); When test your code in non-autograder environment, you can use srand(time(0));

You need to include library cstdlib for srand and ctime for time.

Competition::Competition(Hare \*rabbit, Tortoise \*tor,

**int** length)

{

srand(1);

**int** rabbitPatternSize = rabbit->getPatternLength();

**int**\* rabbitPattern = rabbit->getPattern();

hare = **new** Hare(rabbitPattern, rabbitPatternSize,

rabbit->getPosition());

//TODO:

//initialize data member tor.

rd = **new** Road(length);

}

**Warning**:

* If data member is also called rabbit, to avoid confusing with given parameter rabbit, use this->rabbit to represent data member rabbit.
* Use similar notation to distinct data member tor and given parameter tor.

1. How to define start method?

Warning: to avoid different running results, so that the submission can be graded using autograder, we make the following adjustments.

Besides using srand(1); to pick a specific sequence of random integers, we start hare first in each round of competition (so that hare take a random int before tor, the random integer decides how many blocks each animal runs in each round).

Also, use setw(3) to display round number in 3-digit. If you do not do so, even if your content is correctly, the forma is not matched and will not be able to pass autograder.

Output result (tie, hare wins, tortoise wins) in a line (followed by endl).

Here is a pseudocode.

void Competition::start()

begin

initialize lastBlock as the last block of rd.

initialize round to be 1.

As long as neither animal reaches the lastBlock

begin

move hare

Afterwards, adjust positions of hare

if the position is out the blocks of rd,

if current position is negative,

move it to position 0.

Otherwise, if current position is larger than lastBlock

set current position of hare to be lastBlock.

Move tor and adjust its position if it falls off the road.

mark the position of hare on the Road by 'H'

mark the position of tor on the Road by 'T'

cout << setw(3) << round << " ";

print out the current status of Road

if hare and tor are on the same block

print out "Ouch, Tortoise bites Hare."

Before moving to the next round of Competition,

clear the block that stepped on by hare or tor.

Move to the next round by increasing round by 1.

end

If the game is a tie,

print "It is a tie." in a line (followed by endl).

Otherwise, if hare wins,

print "Yuck, hare wins." in a line (followed by endl).

Otherwise, print "Yay!!! tortoise wins!" in a line (followed by endl).

end