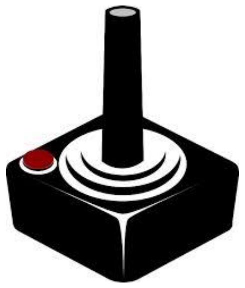
# The College of Informatics Northern Kentucky University & CINSAM

**NKU Summer Programming Workshop 2015**



Project 3:

Part I: Implement the Hare and the Tortoise race as follows.

- Use two int variables: hare and tortoise, to store the current location of each animal in our race course, start them at 0 and the first one that reaches 100 (or greater) wins the game
- Use a while loop to iterate turn by turn
- For each turn, the Tortoise moves randomly from 1 to 3 spaces
- For each turn, there is a 50% chance (1 in 2) that the hare will take a nap (skip its turn)
    - If the hare does not nap, it will move between 1 and 8 spaces
- Output where each animal is after each turn. For instance, the output might look like this:

  The hare is at position 32, the tortoise is at position 36, the tortoise is in the lead!

  The last part (who is in the lead) is optional.

At the end of the race, output who won (notice that if both the hare and tortoise get to 100 or beyond in the same turn, the winner is whichever one has made it further, so the hare wins if hare > tortoise, the tortoise wins if hare < tortoise, otherwise it's a tie). Once you get this working, add the following enhancements one at a time to make the race more interesting. It is suggested (but not necessary) that you do them in this order.

1. There are two streams that run through the path that the race covers. The first stream runs from position 29 to 31 and the second stream runs from 74 to 77. The tortoise can swim across the stream, so if the tortoise lands on say position 30, that's ok. But the hare must jump across. So, for instance, if the hare is at 27 and rolls a 4, the hare must wait because jumping to 31 would put it in the stream. Use in if statement to compare hare+rollAmount to see if it would land in the stream. If so, it does not move (it loses its turn), otherwise you can move the hare.
2. There is a carrot patch from position 48 to 62. If the hare is anywhere in the carrot patch and the hare does not take a nap, then there is a 33% chance (1 in 3) that the hare will decide to eat a carrot during the turn instead of moving.
3. There are black holes at position 44 and 89. If the tortoise or hare land on one of those positions, it is sent back to the start of the race (position 0).
4. Only do this enhancement if you get the entire rest of the game working. If the hare lands on the same spot that the tortoise is current on, the hare must back up 1 spot. If this puts the hare in a stream, back the hare up all the way to before the stream. If this places the hare in a black hole, move the hare to position 0. If the tortoise lands on the same spot that the hare is on, the tortoise moves ahead one spot. If this causes the tortoise land on a black hole, it moves back to 0.

Part II: Implement a program to generate over an over, a random die roll (1-6) and count the number of rolls it takes to roll all 6 values from 1-6. Use 6 boolean variables, r1, r2, r3, r4, r5, r6, all set to false initially. When you roll a 3, see if r3 is false and if so, set it to true and add 1 to a second counter. Repeat until the second counter is 6. Use a while loop to control the iterations until the second counter is 6. Output each die roll and then at the end, output the number of total rolls (the first counter) that it took to get the 6 different values. An enhancement is to require that the 6 numbers be rolled in order. So for instance, if the roll is 4, it doesn't count until r1, r2 and r3 are already true meaning 1, 2 and 3 have already been rolled.