COMP10120 Practical Set 4: Data Types & Pointers

Please read the questions carefully. Name each program based on your student number, the practical set number and question number. For this set (set4), question 1 should be named 1234567s4q1.c where your student number replaces 1234567. All questions that you are submitting can be zipped into a single file called 1234567s4.zip, where 1234567 is your student number and s4 refers to set 4. This zipped file can be submitted via Moodle for grading.

Part 1

1. Write a C Program which prompts the user to enter 4 integers. It should then sum the integers and multiply the integers, storing the results in suitable variables. The program should print each integer value (including the results) and the memory address where each integer is stored. The program should also include a function which squares each of the 4 integers using *pass by reference.*

2. Write a C Program to demonstrate the address structure in arrays. The program should create 2 arrays of different numeric *types (integer and float)* and then print the address of each element of each array to the screen. Format the output so that it is easy to understand. You should notice that the gap between addresses is equal to the size in bytes of the *type* of the array.

Part 2

3. Write a C Program that mimics a two-horse race. Each horse will start at the first grid/space on the race track. The finish line is at 100 hops/spaces away and the first horse and jockey to reach the end wins the race! A **loop** will control how far each horse moves on each iteration by using random number generation according to the rules below (ending when a horse and jockey reaches or passes space 100 on the track):

    A. 50% of the time, the horse will progress at full speed (2 spaces).
    B. 10% of the time, the horse will only progress 1 space.
    C. 10% of the time, the horse will only progress 3 spaces.
    D. 10% of the time, the horse will stall and not move at all.
    E. 20% of the time, the horse will roll backwards 2 spaces.

    To achieve these rules, generate a random integer between 1 and 10. If the integer is between 1 and 5, then use option A (move 2 spaces forward). Do this conditional check for the other possibilities too (e.g. if the number generated is 6 do option C because 6 will occur approximately 10% of the time in the random number generation). Checks need to be included in case a horse happens to go backwards past the first element (they should be reset to the first element space, likewise if the horse goes past the end of the track, move it back to the last position.

    You need to use **variables** to keep track of the position of the horses (i.e. their progress and position on the track). On each iteration of the loop, each horse and jockey will progress 1, 2 or 3 spaces, not progress at all or move back 2 spaces, a **variable** for the each horse will control this.

The program should print the location of each horse at every loop iteration. If they are in the same space, print a T for 'Tied'. Finally, when one horse reaches the last element, print out which car won. If both horses reach the last element on the same iteration of the loop, print out 'Race Tied'.

There is skeleton code to support this program on Moodle (horse_race_skeleton.c), you need to fill in the missing parts which are highlighted with ^^ in the comments. Place particular emphasis on the use of pointers to store the location of the horses and the importance of them in this program.

Portfolio Ideas

1. Adapt the C Program which you wrote in Question 3 to consider an additional horse which has a new set of rules as outlined below.

A. 10% of the time, the horse will progress at super speed (4 spaces).
B. 40% of the time, the horse will only progress 2 spaces.
C. 20% of the time, the horse will progress only 1 space.
D. 10% of the time, the horse will stall and not move at all.
E. 20% of the time, the horse will roll backwards 2 spaces.

Additionally, the race track has a number of jumps/fences. If a horse lands on a jump/fence then they must return to the start of the race track and start again. The jumps are located at positions/spaces 5, 25, 50 and 75. The program should print out 'Horse X hit a fence' when a horse lands on a jump/fence.

2. Write a C Program which prints the size (in bytes) of the following variables to the screen. Depending on your system, you may get different answers to other students.

- char
- int
- long
- long long
- double
- long double
- float
- an array of integers
- a pointer to an array