CSE 425 Studio: Data Types I

These studio exercises and the next ones are intended to acquaint you with ideas and techniques for managing programming language data types, which you will do in a multi-paradigm programming language (C++).

In this studio you will again work in self-selected groups of ~3 people, and will report the results of your work in an email to the course account. Students who are more familiar with the material are encouraged to partner with and help those who are less familiar with it, and asking questions of your classmates and professor during the studio sessions is highly encouraged as well. There are also links on the main course web page to helpful web sites for these exercises, which you also are encouraged to use as resources.

Please record your answers as you work through the following exercises. After you have finished please send an email containing your answers to the required exercises, and to any of the enrichment exercises you completed, with "Data Types Studio I" in the subject line, to the cse425@seas.wustl.edu course account. The enrichment exercise is a chance to dig deeper into the material, especially if you breeze through the required ones.

PART I: REQUIRED EXERCISES

1. Form a team of ~3 people and write down your names as the answer to this exercise.

2. Open up Visual Studio 2013 and create a new project (for example, **`DataTypesStudioI`** or something similar that identifies which studio this is for). Modify the main function signature so that it matches the standard portable (e.g., between Windows and Linux) main function signature (and variable names) for a C++ program: **`int main (int argc, char * argv[])`**.

Inside your main function, declare two variables, of types **`char`** and **`int`** respectively, and initialize both of them with the value **`0`**. Use the C++ sizeof operator to determine how many bytes are used to represent each of those types on the machine where you are working, and print out those representational sizes. Then write a simple loop that (1) iterates through all the possible values that the **`char`** variable can take on (until its value again reaches **`0`**), and (2) records the largest and smallest of its values and after the loop has finished prints them out. Based on your observations, and as the answer to this exercise, predict the largest and smallest values that the **`int`** variable can have on that machine, and explain your reasoning for that prediction.

3. Above your main function, declare an enumerated type (using **`enum`**) for the days of the week. In your main function declare a variable of the enumerated type. Try to increment and decrement its value, determine its size, iterate through all its possible values, print out its minimum and maximum values, and any other operations that allow you to compare and contrast it with the **`int`** and **`char`** types you declared in the previous exercise. As the answer to this exercise describe whether (and if so how) the enumerated type differs from the **`int`** and **`char`** types, as well as any ways you noticed in which they are similar.

4. Inside your main function, declare six variables, of types **`int`** and **`const int`** and **`long`** and **`const long`** and **`double`** and **`const double`** respectively, and initialize them with different values. Attempt to assign each of them to each of the others, and for those assignments that were allowed print out the values of the left-hand-side variables before and after assignment. As the answer to this question, please say which assignments were allowed, what forms of type conversion (if any) occurred in them, and what the values were before and after each of them.

5. Repeat the previous exercise, but use the C++ **`static_cast`** and **`const_cast`** operators to perform explicit conversions between the types rather than the implicit ones on which the assignments relied in the previous exercise. For each assignment that is allowed using those explicit conversion, again print out the values of the left-hand-side variables before and after assignment. As the answer to this question, please say which assignments were allowed, what forms of type conversion (if any) occurred in them, and what the values were before and after each of them.

6. In your main function, declare a C-style string (e.g., **`"hello, world!"`**) and initialize a variable of type **`char *`** to point to it. Also declare and initialize an **`int`** variable and then declare and initialize a variable of type **`int *`** to point to it. Insert the pointers into the standard output stream, and also insert the results of using the C++ **`reinterpret_cast`** operator to convert each of them to type **`void *`**. As the answer to this exercise please summarize what you saw and what that tells you about the overloading of the insertion operator for the standard output stream.

PART II: ENRICHMENT EXERCISE (Optional, feel free to skip or try variations that interest you)

7. Use the sizeof operator to examine the sizes of different objects (e.g., of different types from the previous exercises), predefined types, and arrays of objects and predefined types. As the answer to this exercise describe the sizes of the different types you observed, and indicate whether or not those types are represented efficiently in memory.