CSE 4714 / 6714 – Theory and Implementation of Programming Languages
Lab 3 – Vectors and Classes

Vectors are used in C++ when a resizable list data structure is needed.

https://www.cplusplus.com/reference/vector/vector/

As all data structures (array, vector, set, etc.) in C++, each vector can only hold one data type. That data type is specified when the vector is declared. For example, a vector that holds integers would be declared as

```
vector<int> mylist;
```

How can we store items of different types in the same vector? For example, in the TIPS Interpreter we will need to store a list of statements within a compound statement. But the statements in the list will not be the same type: some are `assignment` statements, some are `if` statements, some are `read` statements, etc.

One way to accomplish this goal is to use pointers and inheritance. In this lab you will create a set of classes that hold data values: strings, integers, and floating-point numbers. Each of these classes will be derived (inherit from) a common superclass: `DataNode`. Then we can declare a vector that contains pointers to `DataNode` instances.

```
vector<DataNode*> aVector;
```

Because each of our `DataNode` subclasses (`StringNode`, `IntegerNode`, `FloatNode`) are `DataNodes`, we can add pointers to string nodes, integer nodes, and float nodes to the vector.

Instructions

Complete the `StringNode`, `IntegerNode`, and `FloatNode` classes. Each class requires constructor, destructor, and `printTo` methods. Each constructor should accept an argument and save that value within the node. The `printTo` method is an example of *double dispatch*:

https://en.wikipedia.org/wiki/Double_dispatch

Once the data node classes are complete, use the comments within `driver.cpp` as instructions on what functionality should be added to the `main` method.

Use the files in `Lab_3_Canvas_Files.zip` as a starting point for your program.

Output

The expected output is shown in the `output.correct` file.

HINT: Use `diff` to compare the output of your program to the expected output file. This is much easier than trying to compare the files manually.

Written Report

Write a report about your experience. The report should cover the following items:

- Describe, in a few sentences, how the derived classes in this assignment allow a list to hold different data types.
- Describe, in your own words, *double dispatch*. In this assignment, when is double dispatch used? How is it implemented?
- Describe, in a few sentences, any sources you consulted during your development of this assignment.
- In addition to the requirements specified above, the report should be neat, well structured, grammatically correct, and use words that are spelled correctly.

List your *name*, *netID*, and the *date* at the top of your report. Otherwise, the report may use any format and fonts. The report does not have any specific length requirement. Completeness is valued more than length, and excessive length could result in a lower grade.

Deliverable

The deliverable is a zip file of the files needed to build and execute your program. Create a zip file named `*netid*_lab_3.zip` and upload that file to the assignment. Do not include any files generated by the `makefile` in your submission. For example, your submission should not include any `.o` or executable files.