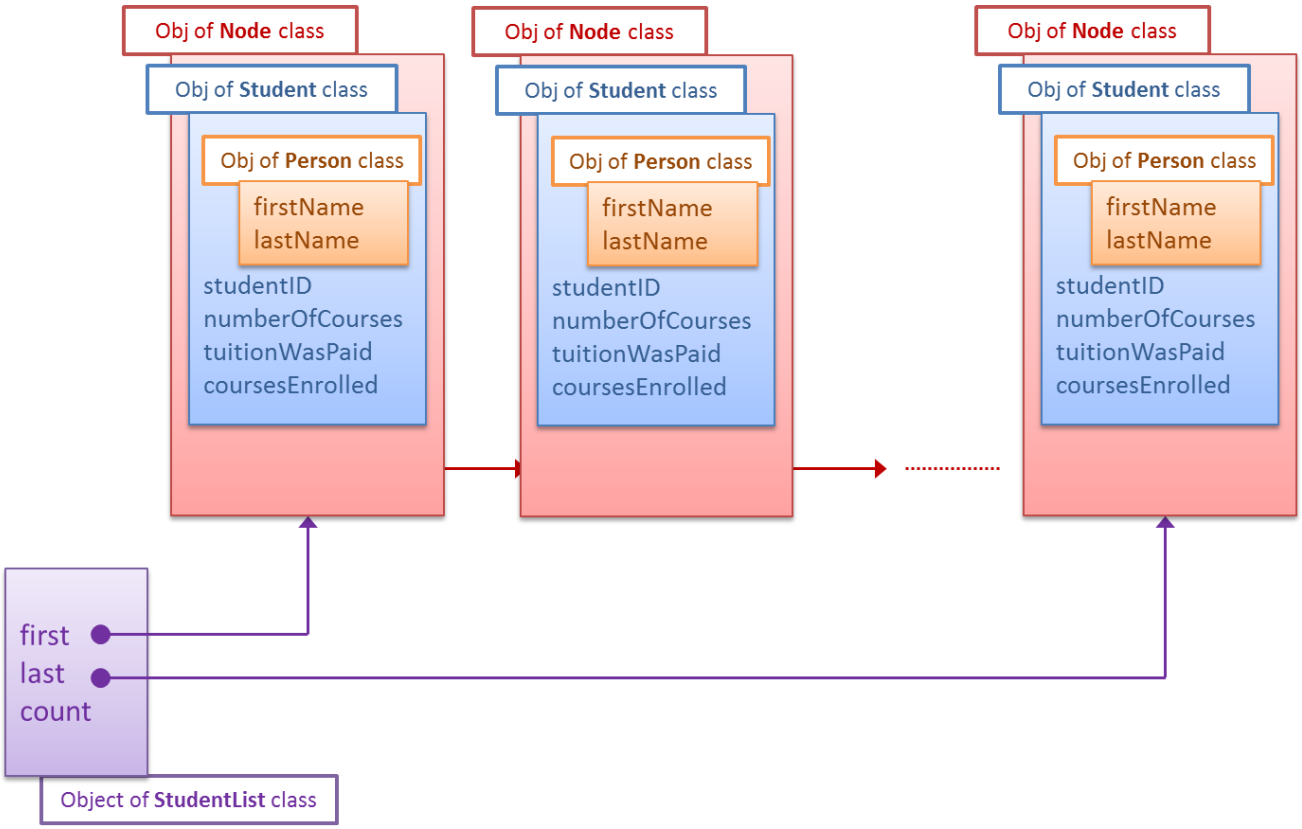
Project 1 – Grade Report (Part D)

For this part of the project, you will complete the class **StudentList** that creates a **singly-linked list** of nodes containing objects of the class **Student** and a pointer to the next node.



Add to your project the following files:

- StudentList.h
- StudentList.cpp
- student_data.txt
- InputHandler.h
- Main.cpp (this will replace your old Main.cpp file)

StudentList.h

The **StudentList.h** file already includes a class **Node** that creates objects containing a **Student** object, **student**, and a pointer **next** to point to the next node. The file also includes the partial definition of the class **StudentList**, which creates objects that contain a pointer **first** to point to the first node in the list, a pointer **last** to point to the last node in the list, and an int **count** to keep track of the number of nodes in the list.

You will need to implement the following member functions:

- **Default constructor**
  - Initializes all member variables.

- Function **addStudent**
  - **Parameter:** an **object** of the **Student** class.
  - Creates a new node, inserts the Student object into the node, and adds the node to **front** of the list.
  - Do not forget that you have a pointer that points to the last node of the list!

- Function **getNoOfStudents**
  - Returns the number of nodes in the list.

- Function **printStudentByID**
  - **Parameters (in this order):** an **int** that stores the student's ID number, a **double** that stores the tuition rate.
  - It searches through the list for a student whose ID matches the one passed by the parameter and then it prints the student information by calling the function **printStudentInfo** of the **Student** class.
  - If the list is empty, output an **error** message, "List is empty."
  - If the ID was not found, output a message, "No student with ID # found in the list." where # is replaced by the ID number passed by the parameter.

- Function **printStudentsByCourse**
  - **Parameter:** a **string** that stores a course number.
  - It prints information about each student that is enrolled in the course passed by the parameter. It calls the functions **isEnrolledInCourse** of the **Student** class to check if the student is enrolled in the course, and then calls the function **printStudentInfo** of the **Student** class.
  - **NOTE:** When traversing the list, check if the student is enrolled in any course at all, before calling other functions.
  - If the list is empty, output an **error** message, "List is empty."
  - If none of the students is enrolled in the course, output a message, "No student enrolled in ***" where *** is replaced by the course number passed by the parameter.

- Function **printStudentByName**
  - **Parameter:** a **string** that stores a last name.
  - It searches through the list for a student whose last name matches the one passed by the parameter and then it prints the student information by calling the function **printStudentInfo** of the **Student** class.
  - If the list is empty, output an **error** message, "List is empty."
  - If the ID was not found, output a message, "No student with last name *** is in the list." where *** is replaced by the last name passed by the parameter.
  - Do **not** assume that there is only one student with the given last name.

- Function **printStudentsOnHold**

- **Parameter:** a **double** that stores the tuition rate.
- It searches through the list for students who did not pay the tuition and prints information for each student by calling the function **printStudentInfo** of the **Student** class.
- Use functions **isTuitionPaid** and **billingAmount** of the **Student** class to find the values you need.
- If the list is empty, output an **error** message, "No students in the list."
- If there are no students on hold, output a message, "There are no students on hold."

- Function **printAllStudents**
  - **Parameter:** a **double** that stores the tuition rate.
  - It traverses the list to print out information about all students by using the function **printStudentInfo** of the **Student** class.

- Function **destroyStudentList**
  - It deletes **each node** in the list and re-sets all member variables to their original values.

- **Destructor**
  - Calls the function **destroyStudentList**.

StudentList.cpp

In here, you need to write the implementation of the functions declared in the StudentList class definition.

student_data.txt

The **student_data.txt** file should be placed in the **Resource Files** folder of your project. The file contains a list of students to add to the list your program creates. The first number in the file is the **tuition rate** for each unit, which in this case is $345.00. This is followed by student data according to the following format:

```
Samantha Parrington 456987 Y 3        => First name/last name/ID/tuition paid/# of classes
ComputerSci CSC201 4 A                => Course name/course number/credits/final grade
Mathematics MTH345 4 A
Biology BIO234 4 A
```

The file ends with the word "**END**" to stop the loop when reading the data.

InputHandler.h

The **InputHandler.h** reads data from the text file and inserts it in the Student list. It first checks if the file is available and the data can be read; if not, it will terminate the program.

The function **createStudentList** creates objects of class Students and objects of class Course (which are going to be inserted in the vector that each Student object holds to store the classes each student is enrolled in. After creating each Student object, it stores the object in the list by calling the function addStudent of the class StudentList.

Although the implementation of this file is complete, do NOT dismiss it! Pay careful attention to the function createStudentList to understand how everything is inserted in the list.

Main.cpp

The **Main.cpp** is partially completed. You will be adding another section next time. For now, you have a menu that allows the user to select one of the following:

Print all students
Print student information
Search student by last name
Print students by course
Print students on hold
To exit

This file is also completed, but do trace it to see what it does and how it connects everything.

Output (folder)

In the **Output folder** you will find the **executable** file that shows you how the output should look.

Test your project by making all selections. No need to turn in. You will be adding another section next time.