Criterion C

Introduction: I used Netbeans (and it's prepackaged Swing tools) in the creation of the code for the relevant classes and GUI elements for this project, in addition to using it as my main IDE.

Summary List of All Techniques

- Parameter Passing
- For loops
- While loops
- Nested loops(?)
- Method returns value
- 2D arrays
- Polymorphism
- Inheritance
- Encapsulation
- Parsing data to another data type
- Compound statements
- Flags
- GUI tabs
- ArrayList's
- Abstract classes
- Constructors
- Accessors
- Superclasses and Subclasses

Program Structure

What: Inheritance is implemented to initialize an hierarchy by extending (the abstracted Superclass) GatePass. Subclasses are different types of GatePass's, which are SeniorGatePass and NormalGatePass.

Why: Inheritance was implemented out of a need to be able to support polymorphism in that my MainGUI class contains a ArrayList that is composed of both SeniorGatePass and NormalGatePass. This is possible because they are both GatePass's. In order to use inheritance more effectively, I split up the structure of the program into 3 different template classes in the superclass, subclass, subclass format, which would translate to GatePass, NormalGatePass, and SeniorGatePass, all of which were "polymorphized" and "ArrayListed" in the MainGUI class to distinguish between the former and the latter gate passes. This was supported by the abstraction of the GatePass class, which "standardised" the definition of what constituted a GatePass in the relevant subclasses.

Data Structures

Here are the data structures that I have included in my program:

- ArrayList
    - Dynamic sizing
    - Efficiency
        - Will consume only as much memory is needed
    - Supports custom objects (in this project, I used an ArrayList of students, normalPass, and seniorPass. Included to highlight high degree of customization)

Main Unique Algorithms

```
if(the senior gate pass checkbox is ticked){
      add a new student that uses a senior gate pass,
      and get(name,
              grade level,
              ID number,
              comments,
              senior gate pass number,
              and a true value (boolean)
              )
      }

else{
      add new student thaat has a normal gate pass,
      and get(name,
              grade level,
              ID number,
              and comments
              )
      }

then clear out all text fields, and set checkbox to false (boolean)
```

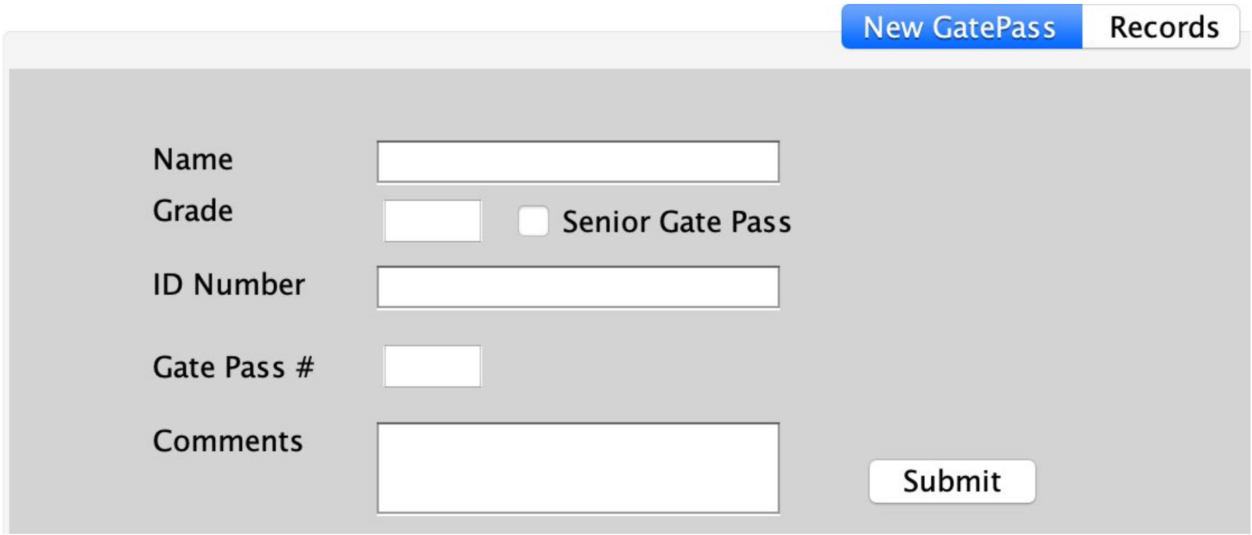
This algorithm is used to distinguish in between the two different types of gate passes. The “decisive” factor here that allows the program to work proper is checking whether the “senior gate pass checkbox” is ticked. If it is, get the information associated with a normal gate pass in addition to the senior gate pass number. If not, then just get the information associated with a normal gate pass. At the end of either conditional statement, it will always clear out the text fields and reset the checkboxes value.

User Interface/GUI Work

What:

- Text Labels
- Text Fields & Text Panes
- jCheckBox
- jButton
- jTable
- jPanel

Why: A GUI was chosen for this specific task because of the client's needs (justification was that they needed to view data and search for data in a efficient manner)



Software Used

What:

- Netbeans IDE

Why: The reason Netbeans was chosen for this task is because building GUIs in Netbeans is extremely simple and intuitive, and also because it provided a "easy" way for me to view and organize my code.

TOTAL WORD COUNT: 325