CSCI 1301 Final Project

By: Wilson Alldredge, Luke Tierney, & Elianna Vallianatos

Student Info Class

Variables:

```
public class studentInfo
{
    //declaring variables for studentInfo class
    private String studentID;
    private String firstName;
    private String lastName;
    private String birthday;
    private String phoneNumber;
    private String email;
    private String gender;
    private String address;
```

- These are all for our student's information
  - All private so they are accessed by this class only

Get Methods:

```
//set and get method for studentID variable
public void setStudentID(String a)
{
   studentID = a;
}

public String getStudentID()
{
   return studentID;
}

//set and get method for firstName variable
public void setFirstName(String b)
{
   firstName = b;
}

public String getFirstName()
{
   return firstName;
}

//set and get method for lastName variable
public void setLastName(String c)
{
   lastName = c;
}
```

- These are our set & get methods
  - Redundant to list all of them
  - Organized it this way to compartmentalize them easier

Student Info Constructor:

```
public studentInfo(String studentID, String firstName, String lastName, String birthday, String mobileNumber, String email, String gender, String address)
{
    this.studentID = studentID;
    this.firstName = firstName;
    this.lastName = lastName;
    this.birthday = birthday;
    this.mobileNumber = mobileNumber;
    this.email = email;
    this.gender = gender;
    this.address = address;
}
```

- Constructor for studentInfo class
  - Used to initialize values of all the strings
  - .this keyword allows parameters passed to correspond to the initialized variables of the class

studentSampleData:

```
//creating list of the sample data and adding objects to them
ArrayList<studentInfo> studentSampleData = new ArrayList<>();
studentSampleData.add(new studentInfo("202301", "John", "Doe", "03/12/1995", "123-456-7890", "john@email.com", "Male", "123 Main St, Atlanta, GA"));
studentSampleData.add(new studentInfo("202302", "Alice", "Johnson", "08/25/1997", "987-654-3210", "alice@email.com", "Female", "456 Elm St, Atlanta, GA"));
studentSampleData.add(new studentInfo("202303", "Bob", "Smith", "11/5/1996", "555-123-7890", "bob@email.com", "Male", "789 Oak St, Atlanta, GA"));
```

- ArrayList<studentInfo> studentSampleData = new ArrayList<>() → a new array list that stores objects of studentInfo class
- studentSampleData.add adds the new information

Iteration:

```
//iterates over the list
for(studentInfo studentInfo: studentSampleData)
{
   System.out.format("| %-10s | %-10s | %-9s | %-10s | %-13s | %-15s | %-6s | %-25s |", studentInfo.getStudentID(), studentInfo.getFirstName(), studentInfo.getLastName(), studentInfo.getBirthday(), st
   System.out.println();
}
line();
```

- Uses enhanced for loops again
- Loops iterate over each studentInfo object in the studentSampleData list
- .format %-10s, etc corresponds with order of methods. %-10 = studentInfo.getStudentID(), etc
- .get methods retrieve the corresponding values for each student

Public Method:

```
//overrides
public String toString()
{
   return String.format("| %-10s | %-10s | %-9s | %-10s | %-13s | %-15s | %-6s | %-25s |", getStudentID(), getFirstName(),
}
```

- .format creates & returns formatted string
- %-10s & etc correspond to the values returned by the methods and adds appropriate spacing (getStudentID(), etc.)

Student Info Header Method:

```java
//this creates the header
public static void header()
{
   System.out.println("-------------------------------------------------------------------------------------------------------------------------");
   System.out.printf("| %-10s | %-10s | %-8s | %-10s | %-12s | %-15s | %-6s | %-24s  |", "Student ID", "First Name", "Last Name", "Birth Date", "Mobile Number", "Email", "Gender", "Address");
   System.out.println();
   System.out.println("-------------------------------------------------------------------------------------------------------------------------");
}
```

- Creates the header method
  - Printf:
    - Sets the attributes' widths
      - %-10s = Student ID, etc
- Dashes are used for organization
- Created a method to call everytime a header is needed (eliminates repeated code)

Dashed Line Method:

```
//prints a dashed line
public static void line()
{
   System.out.println("------------------------------------------------------------------------------------------------------------------------");
}
```

- Created a method to call every time a dashed line is needed (eliminates repeated code)

Header/Dashed Line Method:

```
//method that prints out the students by mobile number
public static void printStudentsByMobileNumber(ArrayList<studentInfo> students, String targetMobileNumber)
{
    header();
    for (studentInfo student : students)
    {
        if(targetMobileNumber.equalsIgnoreCase(student.getMobileNumber()))
        {
            System.out.println(student);
        }
    }
    line();
}
```

```
-------------------------------------------------------------------------------------------------------------------------
| Student ID | First Name | Last Name | Birth Date | Mobile Number | Email           | Gender | Address                 |
-------------------------------------------------------------------------------------------------------------------------
| 202301     | John       | Doe       | 03/12/1995 | 123-456-7890  | john@email.com  | Male   | 123 Main St, Atlanta, GA  |
| 202302     | Alice      | Johnson   | 08/25/1997 | 987-654-3210  | alice@email.com | Female | 456 Elm St, Atlanta, GA   |
| 202303     | Bob        | Smith     | 11/5/1996  | 555-123-7890  | bob@email.com   | Male   | 789 Oak St, Atlanta, GA   |
-------------------------------------------------------------------------------------------------------------------------
```

printStudentsByID:

```
//method that prints out the students by student ID
public static void printStudentsByID(ArrayList<studentInfo> students, String targetStudentID)
{
   header();
   for (studentInfo student : students)
   {
      if(targetStudentID.equalsIgnoreCase(student.getStudentID()))
      {
         System.out.println(student);
      }
   }
   line();
}
```

- Defines printStudentsByID
- Takes two parameters: ArrayList<studentInfo> students & String targetStudentID
- Prints out information about students whose ID matches targetStudentID
- : = enhanced loop, simplifying search
- line() prints organizational structure
- header() method is called for organizational structure (previous slide)

Running Code (1):

```
----jGRASP: process ended by user.

----jGRASP exec: java -ea studentInfo
-----------------------------------------------------------------------------------------------------------------------
| Student ID | First Name | Last Name | Birth Date | Mobile Number | Email           | Gender | Address                |
-----------------------------------------------------------------------------------------------------------------------
| 202301     | John       | Doe       | 03/12/1995 | 123-456-7890  | john@email.com  | Male   | 123 Main St, Atlanta, GA |
| 202302     | Alice      | Johnson   | 08/25/1997 | 987-654-3210  | alice@email.com | Female | 456 Elm St, Atlanta, GA  |
| 202303     | Bob        | Smith     | 11/5/1996  | 555-123-7890  | bob@email.com   | Male   | 789 Oak St, Atlanta, GA  |
-----------------------------------------------------------------------------------------------------------------------



Enter the ID to print that Student's information: 202301
-----------------------------------------------------------------------------------------------------------------------
| Student ID | First Name | Last Name | Birth Date | Mobile Number | Email           | Gender | Address                |
-----------------------------------------------------------------------------------------------------------------------
| 202301     | John       | Doe       | 03/12/1995 | 123-456-7890  | john@email.com  | Male   | 123 Main St, Atlanta, GA |
-----------------------------------------------------------------------------------------------------------------------
```

- Displays given student information
- Searches for a student using printStudentsByID method

printStudentsByFirstName:

```
//method that prints out the students by first name (not case-sensitive)
public static void printStudentsByFirstName(ArrayList<studentInfo> students, String targetFirstName)
{
   header();
   for (studentInfo student : students)
   {
      if(targetFirstName.equalsIgnoreCase(student.getFirstName()))
      {
         System.out.println(student);
      }
   }
   line();
}
```

- Defines printStudentsFirstName
- Takes two parameters: ArrayList<studentInfo> students, & String targetStudentID
- Prints out information about students whose first name matches targetFirstName
  - : = enhanced loop, simplifying search
- equalsIgnoreCase ignores case sensitivity
- line() method is called & prints organizational structure
- header() method is called for organizational structure

Running Code (2):

```
Enter the first name to print that Student's information: John
-----------------------------------------------------------------------------------------------------------------------
| Student ID | First Name | Last Name | Birth Date | Mobile Number | Email           | Gender | Address                |
-----------------------------------------------------------------------------------------------------------------------
| 202301     | John       | Doe       | 03/12/1995 | 123-456-7890  | john@email.com  | Male   | 123 Main St, Atlanta, GA  |
-----------------------------------------------------------------------------------------------------------------------
```

- Displays information based on first name

printStudentsByLastName:

```
//method that prints out the students by last name (not case-sensitive)
public static void printStudentsByLastName(ArrayList<studentInfo> students, String targetLastName)
{
   header();
   for (studentInfo student : students)
   {
      if(targetLastName.equalsIgnoreCase(student.getLastName()))
      {
         System.out.println(student);
      }
   }
   line();
}
```

- Does all the same things prior but searches for last name instead

Running Code (3):

```
Enter the last name to print that Student's information: smith
-----------------------------------------------------------------------------------------------------------------------
| Student ID | First Name | Last Name | Birth Date | Mobile Number | Email           | Gender | Address                |
-----------------------------------------------------------------------------------------------------------------------
| 202303     | Bob        | Smith     | 11/5/1996  | 555-123-7890  | bob@email.com   | Male   | 789 Oak St, Atlanta, GA  |
-----------------------------------------------------------------------------------------------------------------------
```

- Displays information based on last name

printStudentsByBirthday:

```
//method that prints out the students by birthday
public static void printStudentsByBirthday(ArrayList<studentInfo> students, String targetBirthday)
{
   header();
   for (studentInfo student : students)
   {
      if(targetBirthday.equalsIgnoreCase(student.getBirthday()))
      {
         System.out.println(student);
      }
   }
   line();
}
```

- Does all the same things prior but searches for birthday instead

Running Code (4):

```
Enter the birthday to print that Student's information (MM/DD/YYYY): 08/25/1997
-----------------------------------------------------------------------------------------------------------------
| Student ID | First Name | Last Name | Birth Date | Mobile Number | Email           | Gender | Address          |
-----------------------------------------------------------------------------------------------------------------
| 202302     | Alice      | Johnson   | 08/25/1997 | 987-654-3210  | alice@email.com | Female | 456 Elm St, Atlanta, GA  |
-----------------------------------------------------------------------------------------------------------------
```

- Displays information based on birthday

- The code continues to do the same things except search for specific things instead:
  - Phone number
  - Email
  - Gender
  - Address

```java
//method that prints out the students by mobile number
public static void printStudentsByMobileNumber(ArrayList<studentInfo> students, String targetMobileNumber)
{
   header();
   for (studentInfo student : students)
   {
      if(targetMobileNumber.equalsIgnoreCase(student.getMobileNumber()))
      {
         System.out.println(student);
      }
   }
   line();
}

//method that prints out the students by email (not case-sensitive)
public static void printStudentsByEmail(ArrayList<studentInfo> students, String targetEmail)
{
   header();
   for (studentInfo student : students)
   {
      if(targetEmail.equalsIgnoreCase(student.getEmail()))
      {
         System.out.println(student);
      }
   }
   line();
}
```

```java
//method that prints out the students by gender (not case-sensitive)
public static void printStudentsByGender(ArrayList<studentInfo> students, String targetGender)
{
   header();
   for (studentInfo student : students)
   {
      if (targetGender.equalsIgnoreCase(student.getGender()))
      {
         System.out.println(student);
      }
   }
   line();
}

//method that prints out the students by gender (not case-sensitive)
public static void printStudentsByAddress(ArrayList<studentInfo> students, String targetAddress)
{
   Scanner scan = new Scanner(System.in);
   String userID = scan.nextLine();
   header();
   for (studentInfo student : students)
   {
      if (targetAddress.equalsIgnoreCase(student.getAddress()))
      {
         System.out.println(student);
      }
   }
   line();
}
```

Running Code (5):

```
Enter the gender to print that student's information (male/female): male
-----------------------------------------------------------------------------------------------------------------------
| Student ID | First Name | Last Name | Birth Date | Mobile Number | Email           | Gender | Address                |
-----------------------------------------------------------------------------------------------------------------------
| 202301     | John       | Doe       | 03/12/1995 | 123-456-7890  | john@email.com  | Male   | 123 Main St, Atlanta, GA |
| 202303     | Bob        | Smith     | 11/5/1996  | 555-123-7890  | bob@email.com   | Male   | 789 Oak St, Atlanta, GA  |
-----------------------------------------------------------------------------------------------------------------------
```

- Displays information based on gender
  - Both males are printed

User Input:

```
//prints out all student info based on user ID input
System.out.print("Enter the ID to print that Student's information: ");
String userID = scan.nextLine();

if(userID.equals(studentSampleData.get(0).getStudentID()))
{
   header();
   System.out.println(studentSampleData.get(0));
   line();
}
else if(userID.equals(studentSampleData.get(1).getStudentID()))
{
   header();
   System.out.println(studentSampleData.get(1));
   line();
}
else if(userID.equals(studentSampleData.get(2).getStudentID()))
{
   header();
   System.out.println(studentSampleData.get(2));
   line();
}
else
{
   System.out.println("ID is not in database.");
}


System.out.println("\n\n");
```

- User is prompted to enter a student ID
- Afterwards, the code checks each student ID individually
- If it is searched for and found, information is printed
  - If not, it tells the user “ID is not in database”
- header() & line() are printed after as well

Class information

For our class information, we used the same code structure as the student information. The only things that we changed were the parameters

Problems we faced

Problem #1

```java
//creating the student info array
int r = 14;
int c = 7;
String[][] infoArray = new String[r][c];
```

Two dimensional arrays seemed promising at first, but shortly after we realized that you can not individually format each element.

```java
//print the array
for(int i = 0; i < infoArray.length; i++)
{
    for(int j = 0; j < infoArray[i].length; j++)
    {
        System.out.printf(format:"%-7.20s", infoArray[i][j] + " / ");
    } System.out.println();
}
```

Our output was not as clean as we wanted. Due to this, we decided to use the ArrayList class instead.

Problem #2

- Student information would not print if the user inputted different cases than the original address object
- Used .equalsIgnoreCase() to fix the problem
- User can enter variety of upper case/lower case characters into the string and the information will print

```
//method that prints out the students by gender (not case-sensitive)
public static void printStudentsByAddress(ArrayList<studentInfo> students, String targetAddress)
{
   Scanner scan = new Scanner(System.in);
   String userID = scan.nextLine();
   header();
   for (studentInfo student : students)
   {
      if (targetAddress.equalsIgnoreCase(student.getAddress()))
      {
         System.out.println(student);
      }
   }
   line();
}
```