Platform-based Programming

#4 Generic classes

1) ArrayList<T>

2019년 2학기

Program Output

- Make a Java program that can manage students for a school

```
Enter Operation String! add
PNU James 1
[Name: James, School:PNU, 1학년]
Enter Operation String! add
SNU James 1
[Name: James, School:SNU, 1학년]
Enter Operation String! list
Total School Count: 2
School Name: PNU Student Count: 1
[Name: James, School:PNU, 1학년]

School Name: SNU Student Count: 1
[Name: James, School:SNU, 1학년]
```

```
Enter Operation String! add
SNU Brown 2
[Name: Brown, School:SNU, 2학년]
Enter Operation String! list
Total School Count: 2
School Name: PNU Student Count: 1
[Name: James, School:PNU, 1학년]

School Name: SNU Student Count: 2
[Name: James, School:SNU, 1학년]
[Name: Brown, School:SNU, 2학년]

Enter Operation String! find
James 1
2 found
[Name: James, School:PNU, 1학년]
[Name: James, School:SNU, 1학년]
Enter Operation String! find
James 2
No Student Found with name James and year 2
Enter Operation String! Clear
Enter Operation String! list
Total School Count: 0
```

```
Enter Operation String! add
CNU Kim 3
[Name: Kim, School:CNU, 3학년]
Enter Operation String! add
PNU James 1
[Name: James, School:PNU, 1학년]
Enter Operation String! add
CNU James 1
[Name: James, School:CNU, 1학년]
Enter Operation String! find
James 1
2 found
[Name: James, School:CNU, 1학년]
[Name: James, School:PNU, 1학년]
Enter Operation String! list
Total School Count: 2
School Name: CNU Student Count: 3
[Name: Kim, School:CNU, 3학년]
[Name: James, School:CNU, 1학년]

School Name: PNU Student Count: 1
[Name: James, School:PNU, 1학년]
```

Program Skeleton

```
public class SchoolManagerTest {
   private static Scanner scanner = new Scanner(System.in);
   private static SchoolManager schoolManager = new SchoolManager() ;

   public static void main(String[] args) {
      while ( true ) {
         final OperationKind op = getOperation() ;
         // process for QUIT, INVALID
         switch ( op ) {
         case ADD : {
            Student newStudent = createStudent() ;
            System.out.println(newStudent) ;
            break ;
         }
         case FIND:
            findStudent() ; break ;
         case CLEAR:
            schoolManager.removeAllSchools() ; break ;
         case LIST:
            System.out.println(schoolManager) ; break ;
         }
      }
   }
```

```
[Name: James, School:PNU, 1학년]
```

```
2 found
[Name: James, School:PNU, 1학년]
[Name: James, School:SNU, 1학년]
```

```
Total School Count: 2
School Name: PNU Student Count: 1
[Name: James, School:PNU, 1학년]

School Name: SNU Student Count: 1
[Name: James, School:SNU, 1학년]
```

```
private static Student createStudent() {
    final String schoolName = scanner.next() ;
    final String studentName = scanner.next() ;
    final int schoolYear = scanner.nextInt() ;

    School theSchool = schoolManager.findSchool(schoolName) ;
    if ( theSchool == null )
        theSchool = schoolManager.createSchool(schoolName) ;
    final Student newStudent =
        new Student(theSchool, studentName, schoolYear) ;
    theSchool.addStudent(newStudent) ;
    return newStudent ;
}
```

```
private static void findStudent() {
   final String studentName = scanner.next() ;
   final int schoolYear = scanner.nextInt() ;
   final ArrayList<Student> foundStudents =
      schoolManager.findStudent(studentName, schoolYear) ;

   if ( foundStudents.size() > 0 ) {
      System.out.println(foundStudents.size() + " found") ;
      for ( Student s : foundStudents ) System.out.println(s) ;
   }
   else
      System.out.println("No Student Found with name " +
         studentName + " and year " + schoolYear) ;
}
```

```
// School.java
public class School {
   private final String name ;
   private ArrayList<Student> students = new ArrayList<>() ;

   public School(final String name) { this.name = name ; }
   …
}
```

```
// Student.java
public class Student {
   private final String name ;
   private int year ;
   private final School theSchool ;
   …
}
```

```
// SchoolManager.java
public class SchoolManager {
   private ArrayList<School> schools = new ArrayList<>() ;
   …
}
```

```
// Code fragment about using ArrayList<T>

ArrayList<Student> students = new ArrayList<>() ;

Student s1 = new Student(….) ;
students.add(s1) ;
Student s2 = new Student(….) ;
students.add(s2) ;

Student s = students.get(0) ;

students.clear() ;
```