Platform-based Programming

#3 Classes

1) Class definition
2) constructor
3) Method definition
4) toString(), equals(), hashcode()

2019년 2학기

Program Output

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

```
Enter Command String! add
James 1
Enter Command String! list
School Name: PNU Student Count: 1
[James, 1학년]

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

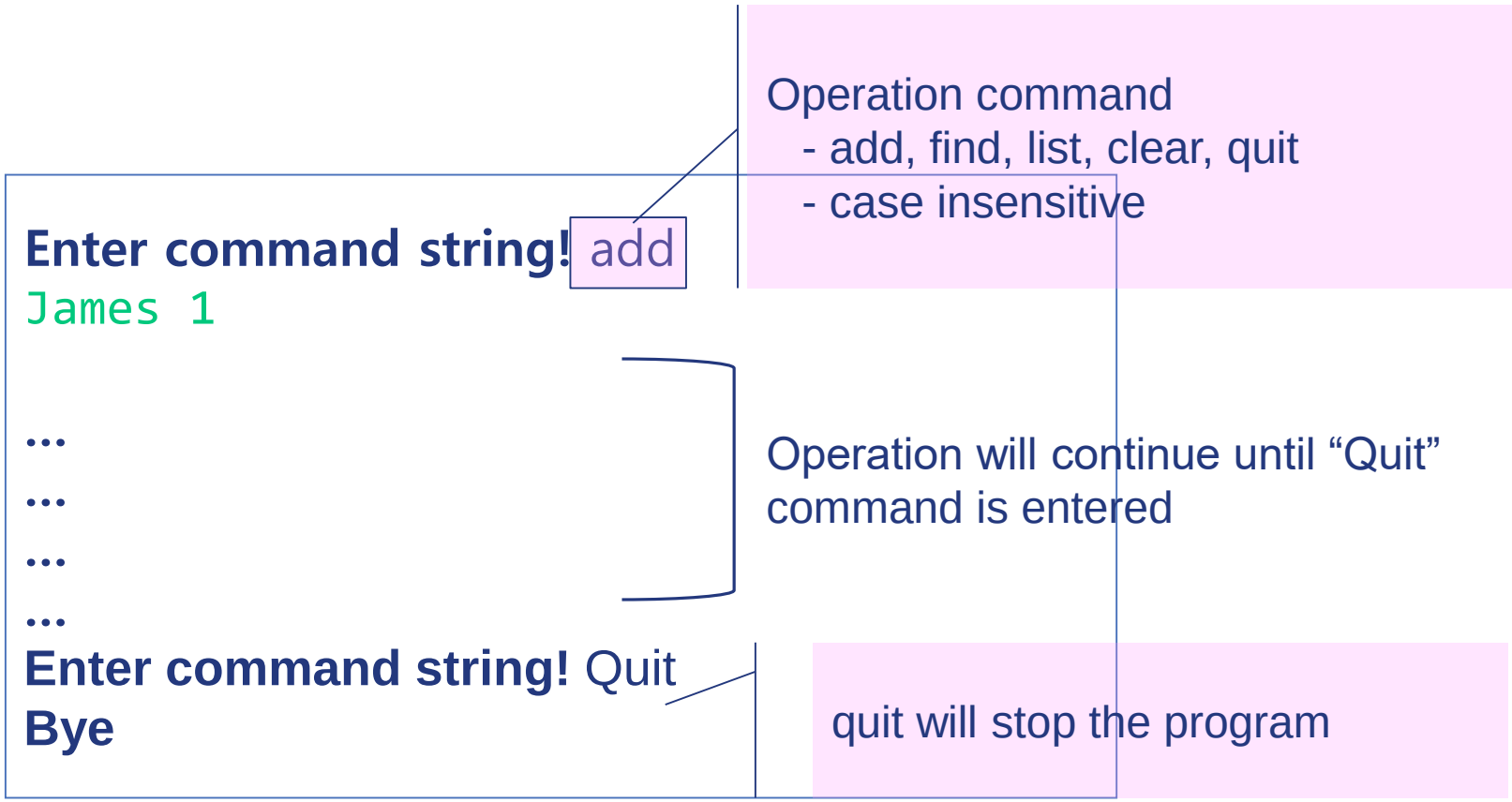
```
Enter Command String! find
Brown 2
[Brown, 2학년]
Enter Command String! find
Brown 1
Student Not Found with name Brown and year 1
Enter Command String! add
Kim 4
Enter Command String! list
School Name: PNU Student Count: 3
[James, 1학년]
[Brown, 2학년]
[Kim, 4학년]

Enter Command String! clear
Enter Command String! list
School Name: PNU Student Count: 0
```

```
Enter Command String! find
Brown 2
Student Not Found with name Brown and year 2
Enter Command String! add
Brown 2
Enter Command String! find
Brown 2
[Brown, 2학년]
Enter Command String! list
School Name: PNU Student Count: 1
[Brown, 2학년]

Enter Command String! find
brown 2
Student Not Found with name brown and year 2
Enter Command String! quit
Bye
```

Problem Description



Program Skeleton

- Write a program by extending the following code!
  - Just add you code! Do not modify/remove the given code

```
// SchoolTest.java
enum OperationKind {
    …
}
public class SchoolTest {
    private static Scanner scanner = new Scanner(System.in);

    public static void main(String[] args) {
        School pnu = new School("PNU", 100) ;
        while ( true ) {
            final OperationKind op = getOperation() ;
            if ( op == OperationKind.QUIT ) {
                System.out.println("Bye") ;
                break;
            }
            if ( op == OperationKind.INVALID ) {
                System.out.println("Invalid Operation!") ;
                continue ;
            }
```

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

   public static void main(String[] args) {
      School pnu = new School("PNU", 100) ;
      while ( true ) {
         // same code in the previous slide

         switch ( op ) {
         case ADD : {
            Student newStudent = createStudent() ;
            pnu.addStudent(newStudent) ;
            break ;
         }
         case FIND: findStudent(pnu) ; break ;
         case CLEAR: pnu.removeAllStudent() ; break ;
         case LIST: System.out.println(pnu) ; break ;
         }
      }
   }
```

```
  private static void findStudent(final School school) {
    final String studentName = scanner.next() ;
    final int schoolYear = scanner.nextInt() ;
    final Student foundStudent =
      school.findStudent(studentName, schoolYear) ;
    if ( foundStudent != null )
      System.out.println(foundStudent) ;
    else
      System.out.println("Student Not Found with name " + studentName +
          " and year " + schoolYear) ;
  }
  private static Student createStudent() {
    final String studentName = scanner.next() ;
    final int schoolYear = scanner.nextInt() ;
    return new Student(studentName, schoolYear) ;
  }
}
```

```
Enter Command String! find
Brown 2
[Brown, 2학년]
```

```
// School.java
public class School {
   private String name ;
   private int limit ;

   private Student[] students ;
   private int studentCount ;

   public String toString() {
      String msg = "School Name: " + name + " Student Count: “
         + studentCount + "\n" ;

      for ( int i = 0 ; i < studentCount ; i ++ ) {
         msg += "\t" + students[i] + "\n" ;
      }
      return msg ;
   }

   // other methods including constructor, equals(), hashCode()
}
```

```
School Name: PNU Student Count: 3
[James, 1학년]
[Brown, 2학년]
[Kim, 4학년]
```

```
// Student.java
public class Student {
   private String name ;
   private int year ;

   // other methods including constructor,
   // toString(), equals(), hashcode()
}
```