객체지향프로그래밍및실습 – 12주차 실습활동지(11월20일)

성명: _____ 학과: _____ 학번: _____

제출방식: 보고서를 pdf 형식으로 제출한다(report12.pdf).

제출기한: 11월 21일(금) 23시59분까지 (지연 제출 0점 처리함, 미완성이라도 반드시 제출하길 바람)

I. Objectives

1. Generic method를 작성하고 활용할 수 있다.
2. Generic class를 작성하고 활용할 수 있다.
3. Type variable의 upper bound를 이해하고 작성할 수 있다.
4. Wildcard를 활용할 수 있다.

II. Exercises (15점)

1. Pair class와 PairTest1 class(첨부파일 Pair.java, PairTest1.java 참조)를 실행해 본 후 다음 물음에 답하시오.

가) PairTest1 클래스 main 메소드 하단의 /**/에서는 Employee 객체(첨부파일 Employee.java 참조)의 배열에서 최소값과 최대값을 출력하고 있다. 이를 위해 Employee 객체의 배열을 매개변수로 받아 salary가 최소인 employee와 최대인 employee를 pair로 반환하는 minmax 메소드를 arryAlg 클래스에 추가하시오(method overloading). 단, type을 제외하고는 기존 코드의 구조를 그대로 이용할 것. [**minmax 코드] [2점]**

```
public static void main(String[] args)
{
    …
    Employee[] emps = new Employee[4];
    emps[0] = new Employee("Kim", 100000.0, 2015, 3, 1);
    emps[1] = new Employee("Lee", 200000.0, 2005, 9, 1);
    emps[2] = new Employee("Hong", 80000.0, 2017, 5, 1);
    emps[3] = new Employee("Park", 150000.0, 2008, 3, 1);

    Pair<Employee> ee = ArrayAlg.minmax(emps);

    System.out.println("min = " + ee.getFirst());
    System.out.println("max = " + ee.getSecond());
}
}
```

나) (가)에서 수정한 코드를 이용하여 PairTest1을 실행해보시오. [**실행결과] [1점]**

다) 아래와 같이 generic method를 만들어 위에서 만든 arrAlg 클래스의 두 개의 minmax 메소드를 교체하고자 한다. 어떤 결과가 나오는가? 왜 그렇게 나오는지 이유를 설명하시오? **[결과 및 이유 설명] [2점]**

```
public static Pair<T> minmax(T[] a)
{
   if (a == null || a.length == 0) return null;
   T min = a[0];
   T max = a[0];
   for (int i = 1; i < a.length; i++)
   {
      if (min.compareTo(a[i]) > 0) min = a[i];
      if (max.compareTo(a[i]) < 0) max = a[i];
   }
   return new Pair<>(min, max);
}
```

라) (다)번의 코드가 정상적으로 작동하기 위해 generic mimax 메소드를 수정하여 실행해 보시오. 힌트) 강의노트 ch08-part2 참조. **[minmax 코드 및 실행결과] [2점]**

2. 아래 PairTest2(첨부파일 PairTest2.java 참조)는 두 명의 Employee 객체의 짝(pair)으로부터 한 명의 Employee 객체를 변경하는 프로그램이다. changePartner() 함수는 Employee pair에서 두 번째 employee를 주어진 parameter로 바꾸는 함수이다. 아래 프로그램을 실행해 본 후 다음 각 질문에 답하시오. (단, 첨부파일 Pair.java, Employee.java, Manager.java를 활용).

```
public class PairTest2
{
   public static void main(String[] args)
   {
        Employee e1 = new Employee("Carl Cracker", 75000, 2007, 12, 15);
        Employee e2 = new Employee("Gus Greedy", 80000, 2000, 1, 15);
        Pair<Employee> p = new Pair<>(e1,e2);
        System.out.println(p.getFirst());
        System.out.println(p.getSecond());

        Employee e3 = new Employee("Sid Sneaky", 100000, 1989, 1,1);

        changePartner(p, e3);
        System.out.println(p.getFirst());
        System.out.println(p.getSecond());
   }
```

```
    public static void changePartner(Pair<Employee> p, Employee e)
    {
        p.setSecond(e);
    }
}
```

가) 위 코드를 실행해보시오. **[답안 포함 안함]**

나) main() 메소드를 아래와 같이 변경한 후 컴파일해보시오. 어떤 문제가 발생하는지 그리고 왜 발생하는지 설명하시오. 힌트 강의노트 ch08-part2 참조 **[문제점 및 이유] [2점]**

```
public static void main(String[] args)
{
    Manager m1 = new Manager("Sarah Smith", 105000, 2007, 12, 15);
    Manager m2 = new Manager("Bobby Brown", 1280000, 2000, 1, 15);
    Pair<Manager> p2 = new Pair<>(m1,m2);
    System.out.println("First = "+p2.getFirst());
    System.out.println("Second = "+p2.getSecond());

    Manager m3 = new Manager("Mike Miller", 200000, 1989, 1,1);
    changePartner(p2, m3);
    System.out.println("First = "+p2.getFirst());
    System.out.println("Second = "+p2.getSecond());
}
```

다) Employee와 Manager 객체의 pair에 정상적으로 모두 동작하도록 bounded type parameter를 이용하여 changePartner()함수를 고치시오. 단, changePartner()는 Employee와 모든 subclass의 pair에 동작해야 한다. 수정한 메소드로 (가)와 (나)의 main 메소드를 모두 실행해보시오. [**changePartner 코드 및 실행결과] [2점]**

3. 아래 Test3 클래스(첨부파일 Test3.java 참조)의 printEmployee는 Employee 및 subclass의 list를 받아 각 원소를 출력하는 메소드이다. 아래 물음에 답하시오. 단, Employee와 Manager 클래스는 문제 2번의 첨부자료 활용.

```
public class Test3
{
    public static void main(String[] args)
    {
        ArrayList<Employee> le = new ArrayList<>();
        le.add(new Employee("Carl Cracker", 75000, 2007, 12, 15));
        le.add(new Employee("Gus Greedy", 80000, 2000, 1, 15));
        le.add(new Employee("Sid Sneaky", 100000, 1989, 1,1));

        printEmployee(le);

        ArrayList<Manager> lm = new ArrayList<>(;
```

```
        lm.add(new Manager("Gang Gamchan", 75000, 2007, 12, 15));
        lm.add(new Manager("Yang Manchun", 80000, 2000, 1, 15));
        lm.add(new Manager("Lee Sunshin", 100000, 1989, 1,1));

        printEmployee(lm);
    }

    public static void printEmployee(ArrayList<Employee> le)
    {
        for(Employee e : le)
            System.out.println(e);
    }
}
```

가) 위 코드를 실행해 본 후 어떤 문제가 있는지 그리고 그 이유가 무엇인지 설명하시오. [**문제점 및 이유 설명] [2점]**

나) (가)번의 문제를 해결하기 위해 wildcard를 이용하여 printEmployee 메소드를 수정하여 실행해보시오. [**printEmployee 메소드 및 실행결과] [2점]**