소프트웨어 프로젝트 Project 1

수업: 소프트웨어 프로젝트 1

학과:컴퓨터정보공학부

학번: 2018202074

학년: 3

이름: 김상우

1. A.Introduction

1 번 문제는 주어진 값들에 대하여 최대공약수와 최소 공배수를 구하는 문제이다. 이 때 input 은 연속해서 들어오게 된다. 해당 코드는 이에 대하여 모든 input 에 대한 최대공약수와 모든 수에 대한 최소공배수를 구해야 한다. 이렇게 구해진 최대공약수와 최소공배수 값들은 input 이 들어올 때 마다 출력되며 그 이전까지 입력되었던 값들 또한 새로운 input 이 들어올 때마다 출력되어야한다.

B. Algorithm

해당 문제를 해결하기 위한 알고리즘은 다음과 같이 진행되었다. 우선 Vector 를 생성한다. 이후 첫 값에 대하여 0 이 들어왔을 경우 종료하게 해주었고, 다른 값이 들어왔을 때는 해당 값을 Vector 에 넣어준 후 GCD, LCN 값 또한 들어온 값으로 초기화한다. 이후 계속해 input 을 받을 수 있도록 무한 반복문으로 들어간다.(while(true)사용) 0 이 들어온다면 반복문을 종료하며 들어온 input 값은 Vector 에 저장된다. 또한 이전 GCD 와의 최대 공약수를 구하고, 이전 LCN 값과의 최소 공배수를 구한다. (이는 벡터 내 모든 값들과 새로운 input 의 GCD, LCN 과 같기 때문에 이렇게 할 수 있다.) 이에 대한 값들을 Format 에 맞게 출력한다. 이러한 방식으로 전체 알고리즘을 구성 하였다. 이때 GCD, LCN 을 구하기 위해 유클리드 호제법을 사용했다. 여기서 유클리드 호제법은 최대공약수를 구하는 방식으로 a>b 라고 했을 경우 a%b=r 이라고 할 때, a 와 b 의 최대 공약수는 b 와 r 의 최대 공약수와 같다는 것을 이용하여 나머지가 0 이 될 때 까지 이를 반복, 나머지가 0 일 때 작은 값(=최대 공약수)을 리턴하여 최대공약수를 구할 수 있었다. 또한, 최소공배수의 경우 a*b/(a 와 b 의 최대공약수)이므로 이 값을 return 해주어 구하였다.

C. Pseudo Code

```
Class gcd(int a, int b){
        If(a%b==0) return b
        Return gcd(b,a%b)
}
Void main(){
        Make vector to save, scanner to take input
        Take first input, if input==0, program terminate
        Save input in vector, Gcd=input, lcn=input
        While(true){
                If(input==0) program terminate
                Save input in vector
```

```
                gcd=do gcd function about gcd, input
                 lcn=lcn*input/(gcd result about lcn and input)
                print vector, gcd, lcn

        }

}
```

D.Result Screen

```
INPUT : 100
Input Numbers : 100
LCN : 100
GCD : 100
INPUT : 200
Input Numbers : 100,200
LCN : 200
GCD : 100
INPUT : 50
Input Numbers : 100,200,50
LCN : 200
GCD : 50
INPUT : 800
Input Numbers : 100,200,50,800
LCN : 800
GCD : 50
INPUT : 70
Input Numbers : 100,200,50,800,70
LCN : 5600
GCD : 10
INPUT : 45
Input Numbers : 100,200,50,800,70,45
LCN : 50400
GCD : 5
INPUT : 19
Input Numbers : 100,200,50,800,70,45,19
LCN : 957600
GCD : 1
INPUT : 0
```

처음 값인 100 을 LCN 과 GCD 값으로 선언하고 vector 안에 넣어주었다. 이 값들을 이후에 들어온 input 값들과 사용, 이전 GCD 와 새 input 의 GCD, 이전 LCN 과 새 input 의 LCN 을 계속해 구하고 이전까지 들어온 값들 또한 잘 출력되는 걸 확인할 수 있다. 이후 0 이 들어오면 프로그램이 끝나는 것 또한 확인할 수 있었다.

E.고찰

해당 코드를 구현하는 것은 다양한 방식으로 구현할 수 있다. 위 방식이 아니더라도 다양한 방식으로 구현이 가능한데, 초기 구상에서는 최소공배수와 최대 공약수를 구하기 위해 반복문을 이용해 계속 값에 곱해주며 각 값으로 나눴을 때 나머지가 0 이 되는 경우를 찾을 생각이었다. 허나 유클리드 호제법을 이용, 재귀를 이용하여 반복문을 사용하는 것보다 더 나은 효율의 결과를 만들어내었다. 또한 lcn 을 별개로 제작하는게 아니라 gcd 의 값을 이용해 효과적으로 메소드의 수를 줄일 수 도 있었다.

2. A.Introduction

   해당 문제는 숫자들을 받고 그 숫자들을 목표로 하는 산을 출력하는 것이다. 해당 input 은 문자열의 형태로 들어오며 이 문자열은 ,를 기준으로 분리된다. (ex 1,3,5,2->1 3 5 2) 시작시 높이를 1 로 한다. 현재 높이와 다른 값이 들어왔을 때는 1 씩 이동하며 현재 높이와 알맞은 값들을 출력하고 현재 높이와 같은 값이 들어왔을 경우, 해당 높이의 값을 한번 더 출력해준다. 위 조건에서 볼 수 있듯이 출력에서의 기울기는 1,0,-1 중 하나가 된다. 또한 마지막 input 값이 들어온 후에는 다시 높이를 1 로 마무리할 수 있게 하여야 한다. 이는 input 으로 0 이 들어올 때 까지 반복된다. 여기서 산은 @로 표현되며 나머지는 하늘로서 ‘로 표현되게 된다.

   B. Algorithm

   우선 문자열을 입력 받을 수 있는 Scanner 를 만든다. 우리는 계속해서 input 을 받아야 되기 때문에 무한 루프를 선언한다. 이후 scanner 로 값을 받고 받은 값이 0 인지 확인하여 0 일 경우 무한루프를 종료하고 프로그램에서 나갈 수 있게 한다. 만약 0 이 아니라면 int 를 받는 vector 를 선언해주고 현재 높이를 0 으로 지정한다. 해당 문자열을 Tokenizer 를 이용해 ‘,’를 기준으로 잘라 ,이 없을 때 까지 높이들을 확보한다. 이 과정에서 vector 를 이용, 현재 값이 문자열로 받아지는 값이 되도록 1 씩 조정하며 이 때의 높이들을 vector 에 하나씩 저장하였다. (ex 1->4 이면 1,2,3,4 가 vector 에 저장된다.) 또한 이 과정에서 height 를 비교하며 가장 높은 height 를 찾아낸다. 위를 반복하여 각각의 경우에서의 산에서의 높이들을 연속되게 만들어 준다. 이렇게 얻어진 vector 값을 바탕으로 반복문을 이용해 해당 값들을 출력한다. 출력되는 높이가 vector(가로) 이하일 경우 @, 초과일경우 ‘를 출력하게 해주어 산의 모양을 표시해준다. 이를 통해 format 에 맞게 산을 구현할 수 있었다.

   C. Pseudo Code

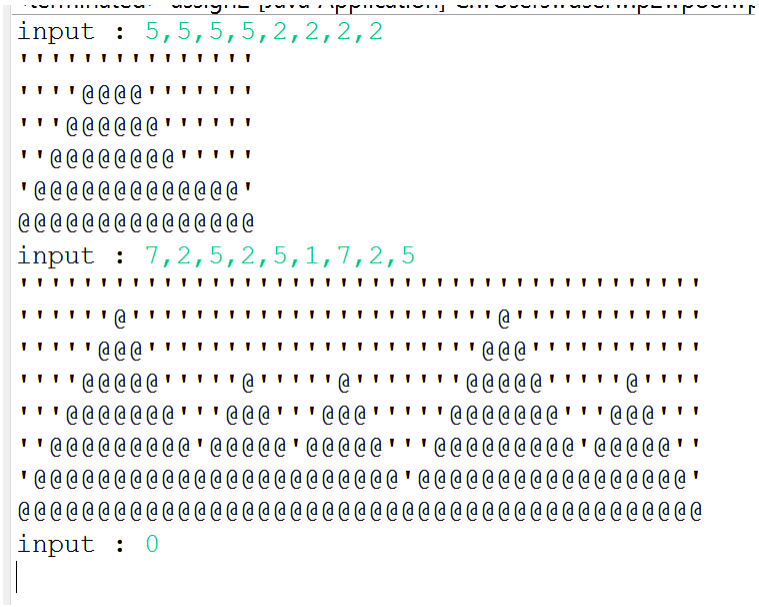
```
Void main(){
        Make scanner to take input
```

```
While(true){
        Take input by user (scanner use)
        If(input=0){
                Break;programterminate;
        }
        StringTokenizer st = new StringTokenizer(input)
        N=0(current hight), high=0(max height)
        Make vector Vec
        While(st.hasMoreTokens()){
                Height=Integer.parseInt(st.nextToken())
                if high<height, high = height
                while(true){
                        if(n>height)? N++ : Vec.add(n)

                                if(n==height)break;
                            }
                    }
            }
            While(true){
                    If(n==1) break;
                    n--; Vec.add(n);
            }
            for(int i=high+1;i>0;i--){
               for(int j=0;j<Vec.size();j++)
                  if(i<Vec.get(j) System.out.print("@")
                  else System.out.print("")
                  }
               }
            }
    }
    Scanner close
}
```

D. Result Screen

```
input : 1,2,3,2,1,2,3,2,1
'''''''''
''@'''@''
'@@@'@@@'
@@@@@@@@@
input : 3,1,3
'''''''''
''@'''@''
'@@@'@@@'
@@@@@@@@@
```



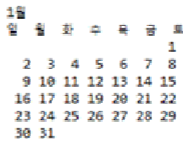
다음과 같이 결과가 나왔다. 시작 높이는 어떤 값이 들어와도 1 로 유지 되는 것을 확인할 수 있으며, string 을 통해 ,을 기준으로 높이들을 받아 결과값을 내고 있음을 확인 할 수 있었다. 또한 기울기 0,1,-1 내에서 연속되게 출력되는 것도 확인할 수 있었다. 대표적으로 맨 처음 input 인 3,1,3 의 경우 처음높이와 마지막 높이가 1 이기 때문에 vector 내에 1,2,3,2,1,2,3,2,1 로 저장되어 최종적으로 input:1,2,3,2,1,2,3,2,1 과 똑같이 출력되고 있는 것을 확인할 수 있다. 또한 같은 값이 여러 번 입력되었을 경우엔 입력된 수만큼 출력되고 있는 것을 input : 5,5,5,5,2,2,2,2 에서 확인 할 수 있었다. 또한 모든 출력에서 최대높이+1 이 세로 길이가 되었음 또한 확인할 수 있었으며 0 이 들어갈 시 종료되는 것도 확인 가능했다.

E.고찰

위 코드를 구현함에 있어서 벡터를 이용해 입력 값들을 연속된 값으로 바꾸고 벡터의 형식으로 저장하여 이후 이 값들을 이용해 2 중 반복문으로 출력했음을 확인할 수 있다. 그러나 Vector 로 변환하는 과정이 컴퓨터에겐 부담일 수 있다. 예를 들어 100 이 입력되었다고 가정하면 Vector 는 1,2…,99,100,99,98…,2,1 을 저장해야 할 것이다. 이를 바탕으로 Vector 의 형식으로 받는 것이 아니라 해당 값들을 연속적으로 변환하는 과정에서 저장이 아닌 출력을 하는 알고리즘으로 수정한다면 컴퓨터에 부담을 덜 주는 방식으로 해결할 수 있을 것이라고 생각한다.

3. A.Intorduction

해당 문제는 Java 를 이용해서 사용자로부터 input 을 받는다. 이때 input 은 년도를 의미한다. 이후 입력된 input 값을 바탕으로 해당 년도의 달력들을 출력하게 된다. 이때 달력은 1 월부터 12 월까지 출력된다. 유의해야 할 것은 달력은 4*3 형태로 출력된다는 것이다. 즉 첫 줄에는 1,2,3 월 두번째 줄에는 4,5,6 월… 이런식으로 출력된다는 것이다. 이 과정에서 input 값은 1900 년부터 2022 년 사이의 값이 들어오게 된다. 달력은 월, 요일, 일에 대한 값을 지녀야 하며 이들은 그림 내 format 과 같은 형태로 출력이 되어야 한다.



B.Algorithm

Import java.util.Calendar 를 이용, class Calendar 를 이용할 수 있게 된다. 이후 3 개의 Month 에 대한 calendar 를 받으며 각 calendar 에 대한 출력시 현재 일, 달의 마지막 일, 현재 요일을 저장하는 변수들을 가질 수 있는 class mycal 를 선언한다. 이 클래스는 생성자로서 각 calendar 에 Instance 를 주고(Calendar.getInstance()사용), 인자를 연도와 Month 를 받아 각 calendar 변수 3 개에 해당 년도 해당 월의 1 일의 값을 저장하고 월과 요일을 format 에 맞춰 출력하며(printf(%-21s)를 이용해 format 정리) 동시에 현재 위치를 1 로 나타내고 해당 달의 마지막 일과 1 일의 요일을 class 내에 저장하는 setcal 메소드를 지닌다.(각각 i,maxDay,weekday 변수로 maxDay 의 경우 this.cal.getActualMaximum(Calendar.DAY_OF_MONTH), weekday 의 경우 Calendar.DAY_OF_WEEK 를 이용해 구한다.)

또한 class mycal 은 (달의 마지막 일, 현재 요일,현재 일,사용할 calendar 번호)를 인자로 받아 현재 줄이 비어있는 마지막 주일 경우 빈 한줄을 출력하고, 해당 주차가 첫 줄일 경우 1 일이 나올 때까지 format 을 맞춰주고 나머지 일 수를 토요일이 될 때까지 출력하며, 해당 주차가 모든 요일에 일이 존재할 경우 모든 일을 토요일까지 출력해주며, 해당주차가 유효한 일을 포함하는 주차이며 마지막 일이 지나 빈 요일을 포함할 경우 유효한 요일까지 일을 출력한 후 남은 일을 공백으로 처리해주어 출력하고 이들이 종료된 후 현재 요일과 일을 선택된 calendar 에 맞추어 저장해주는 printweek 함수를 포함한다.(현재 일을 i, 현재 요일은 weekday, 현재 달의 마지막 날은 maxDay, calendar 선택은 k 로 표현되며 빈 요일에 대한 공백 출력을 위한 값은 weekday%7 이 현재 요일임을 이용, 6-(weekday%7)만큼 공백을

반복하여 출력하였다. 이 method 가 공백을 계속 출력하는 이유는 최종결과의 format 이 4*3 이기 때문에 연속해 3 달을 표현하기 위해 공백을 추가하였다. 각각의 값들은 %3d 를 이용해 format 를 맞춰주었다. 또한 마지막 calendar 를 골라 저장하는 것은 k 값에 따라 this.ik=i; this.weekdayk=weekday 를 이용해 저장해 주었다.

또한 year 를 인자로 받는 printcal 함수가 있다. Printcal 함수는 setcal method 에 해당 년도와 1,4,7,10 을 순서대로 넣고(월과 요일에 대한 format 또한 출력됨.) 각각의 과정에서 printweek 를 이용해 각 calendar 들의 해당 주차의 일들을 출력하고 이를 6 번 반복하여 총 3 개의 달에 대한 출력을 만든다. 이를 4 번 반복하며 해당 년도를 출력하게 된다.

위 함수들을 이용, main 함수에서 scanner 를 통해 년도를 입력 받고, mycal newone 를 만들며 newone.printcal(입력된 년도)를 이용해 달력을 출력한다.

C. Pseudo Code

```
Class mycal{
        Calendar cal1, cal2, cal3(save Calendar)
        Int i1,i2,i3,weekday1,weekday2,weekday3,maxDay1,maxDay2,maxDay3
        (current day to print, current day of week to print, last day of current month)
        Public void set(int year, int month){
                Set day of year and months(month-1,month,month+1) class's calenders
                Print months, day of week for format
                Initialize i1,i2,i3 to 1
                Initialize weekday1/2/3 to week of day of first day of month of year
                Initialize maxDay1/2/3 to last day of month of year

        }
        Public void mycal(){
                this.cal1/2/3 = Calendar.getInstance();
        }
        Public void printweek(int maxDay, int I,int k){
                If(i>maxDay){
                        Print blank for 7 times(blank week)
                }
                For(;i<=maxDay;i++){
                        If(i==1)
                                Print blank during i(current day) is not 1
                        }
                        Print i(current day)
```

```
                    If(i==maxDay){
                        If(week of day!=sunday){
                            Print blanks during  Saturday
                        }
                    }
                    If(week of day=sunday){ Weekday++; i++; break;}
                    Weekday++;
                }
                If(k==1){this.i1=I; this.weekday1=weekday;}
                If(k==2){this.i1=I; this.weekday1=weekday;}
                If(k==3){this.i1=I; this.weekday1=weekday;}
        }
        Public void printcal(int year){
                For(int k=1;k<=10;k+=3){
                    This.setcal(year,k){
                    For(int i=0;i<6;i++){
                        printweek(maxDay1,weekday1,i1,1);
                        System.out.print(“ ”);
                        printweek(maxDay2,weekday2,i2,2);
                        System.out.print(“ ”);
                        printweek(maxDay3,weekday3,i3,3);
                        System.out.print(“ ”);
                    }
                    System.out.println(“ ”);
                }
        }
}
Void main(){
        Scanner scanner = new Scanner(System.in);
        System.out.println(“년도를 입력하세요 : ”)
        Take input(year);
        System.out.println();System.out.println();
        mycal newone = new mycal();
        newone.printcal(input);
}
```

D.Result Screen

```
년도를 입력하세요 :
2022

1월                          2월                          3월
일  월  화  수  목  금  토 |일  월  화  수  목  금  토 |일  월  화  수  목  금  토
                    1           1  2  3  4  5           1  2  3  4  5
 2  3  4  5  6  7  8   6  7  8  9 10 11 12   6  7  8  9 10 11 12
 9 10 11 12 13 14 15  13 14 15 16 17 18 19  13 14 15 16 17 18 19
16 17 18 19 20 21 22  20 21 22 23 24 25 26  20 21 22 23 24 25 26
23 24 25 26 27 28 29  27 28                 27 28 29 30 31
30 31
```

```
4월                                5월                                6월
일  월  화  수  목  금  토 |일  월  화  수  목  금  토 |일  월  화  수  목  금  토
                    1  2    1  2  3  4  5  6  7                1  2  3  4
 3  4  5  6  7  8  9    8  9 10 11 12 13 14    5  6  7  8  9 10 11
10 11 12 13 14 15 16   15 16 17 18 19 20 21   12 13 14 15 16 17 18
17 18 19 20 21 22 23   22 23 24 25 26 27 28   19 20 21 22 23 24 25
24 25 26 27 28 29 30   29 30 31               26 27 28 29 30


7월                                8월                                9월
일  월  화  수  목  금  토 |일  월  화  수  목  금  토 |일  월  화  수  목  금  토
                    1  2       1  2  3  4  5  6                   1  2  3
 3  4  5  6  7  8  9    7  8  9 10 11 12 13    4  5  6  7  8  9 10
10 11 12 13 14 15 16   14 15 16 17 18 19 20   11 12 13 14 15 16 17
17 18 19 20 21 22 23   21 22 23 24 25 26 27   18 19 20 21 22 23 24
24 25 26 27 28 29 30   28 29 30 31            25 26 27 28 29 30
31


10월                               11월                               12월
일  월  화  수  목  금  토 |일  월  화  수  목  금  토 |일  월  화  수  목  금  토
                       1          1  2  3  4  5                   1  2  3
 2  3  4  5  6  7  8    6  7  8  9 10 11 12    4  5  6  7  8  9 10
 9 10 11 12 13 14 15   13 14 15 16 17 18 19   11 12 13 14 15 16 17
16 17 18 19 20 21 22   20 21 22 23 24 25 26   18 19 20 21 22 23 24
23 24 25 26 27 28 29   27 28 29 30            25 26 27 28 29 30 31
30 31
```

위 결과를 보면 format 에 맞게 잘 출력됨을 확인할 수 있었다. 각각의 결과는 상단의 월과 요일을 기재한 것을 제외하면 week 별로 출력된 것이며, 4*3 format 인 만큼 3 달 단위로 출력되었다는 것을 알 수 있다. 이를 정리하면 다음과 같다.

Year 입력됨-> (1-3,4-6,7-9,10-12) month 입력됨->(format 을 위한 요일, 월 출력)->month1,2,3 의 n 번째 week 순서대로 출력. 이를 n=1~n=6 까지 반복->출력되지 않은 month 가 있다면 다음 month 로 이동 후 출력.

E.고찰

해당 문제를 간결하게 해결하기 위해서 반복되는 부분을 최대한 세분화하여 처리하는 것이 합리적이라고 판단하여, week 단위 출력 method 를 제작한 후, 이를 조건에 맞게 반복하게 하여 main 함수가 최대한 간결하게 표현될 수 있도록 하였습니다. 그 결과 class 내에 method 는 생성자를 포함, 4 개가 되었고(각각, class 생성자, class 세팅, 주 단위 출력, 년 단위 출력) 이들이 각자의 method 를 내포하는 형식으로 진행되어 main 에서 class 를 선언하고 method 를 부르는 것만으로 결과가 출력되게 하여 main 함수와 코드가 간결해졌음을 알 수 있습니다. 또한 day 들을 따로 배열로 저장하는 것이 아닌 만큼, 메모리적으로의 성능도 뛰어남을 알 수 있습니다. 기본 method 가 주를 단위로 출력하는 만큼 이후 format 를 바꾸는 데에도 유용할 것이라고 생각합니다.

4. A.Introduction

위 문제는 사용자로부터 1 부터 6 중의 숫자를 입력받아 해당 숫자에 맞는 결과를 진행시켜 주는 코드를 제작하는 것이다 .1 의 경우 Load 로, 지정된 txt 파일의 내용을 읽어오는 것이며 이때 txt 파일이름(Salaries.txt)은 진행되어 있으나 프로젝트의 절대적 경로를 따로 추적할 수 있어야 한다. 이 값들은 이름, 나이, 연봉으로 구성되어 있으며 ‘,’을 기준으로 나뉘어져 있으므로 ‘,’을 기준으로 각 값을 끊어 받아 하나의 객체로서 컬렉션 기반의 데이터로서 저장해야 된다. (txt 파일이 존재하지 않을 경우 에러체크 필요)2.Insert 의 경우 이름과 나이를 입력, 해당 정보를 가진 객체가 없으면 연봉을 입력해 객체를 추가해준다. (있을 시 에러체크)3.Delete 의 경우 이름과 나이를 입력, 해당 정보를 가진 객체가 있으면 연봉을 입력해 객체를 제거해준다.(없을 시 에러체크), 4.Update 의 경우 이름과 나이를 입력 해당 정보를 가진 객체가 있으면 연봉을 입력해 객체의 연봉을 변경해준다.(없을 시 에러체크) 5. Sort 의 경우 Name 을 기준으로 할지 Salary 를 기준으로 할지 정한 후 Name 일 경우 내림차순, Salary 의 경우 오름차순으로 출력한다.(이때 각 값들이 같은 객체끼리는 각각 Salary 기준 내림차순, Name 기준 오름차순으로 비교한다. 나이는 신경 쓰지 않는다.) 6.Save 의 경우 위의 값들을 바탕으로 제작된 결과를 Salaries.txt 라는 이름의 txt 파일로서 저장한다. 이때 이미 해당 이름의 txt 파일이 있을 경우 이를 비우고 새로 생성한다.

*여기서 class 의 변수들은 private 로 표현된다. 위 코드에선 Scanner 와 Collection interface 가 사용된다.

B.Algorithm

Class member 를 선언한다. 이때 member 는 private 으로 String name, int age, int salary 를 받는다. 생성자로 String 과 int 값 2 개를 받아 이들로 각 값들을 초기화하고 private 값을 외부에서 쓸 수 있도록 getname(), getage(), getsalary()를 이용해 외부로 값을 return 할 수 있게 한다. 또한 이 값들을 바탕으로 memstr(),memstr2()를 정의, 각각 console, txt 에 출력될 format 에 맞추어 return 된다.

또한 두 개의 member class object 를 받고 이들의 name 값을 비교해 두 번째 input 이 더 사전적으로 뒤에 있을 경우 1, 앞에 있을 경우 -1 을 return 하며 같을 경우 salary 를 기준으로 두 번째 인자가 더 작을 경우 -1 클 경우 1 을 return 하는 compare method 를 가지는 comname class 를 만든다.(implements Comparator<Member>)

또한 두 개의 member class object 를 받고 salary 를 기준으로 첫번째 인자의 salary 가 더 크면 1 을, 더 작으면 -1 을 return 하고 salary 가 같을 경우 name 을 기준으로 두 번째 input 이 더 사전적으로 뒤에 있을 경우 -1, 앞에 있을 경우 1 을 return 하는 compare method 를 가지는 comint class 를 만든다..(implements Comparator<Member>)

이후main 에서 값들을 저장하기 위해 Vector<Member> sallist를 선언하고(Collection사용)

```
File file = new File(".");
```

```
String rootPath = System.getProperty("user.dir");
File f= new File(rootPath+"\\Salaries.txt");
```

를 이용해 현재 프로젝트의 위치를 지정 후 프로젝트 내 Salaries.txt 를 찾을 수 있도록 하였다. 이후 Scanner 를 이용해 사용자로부터 값을 받을 수 있게 한 후 while(true)를 이용, 사용자가 계속해 input 값을 넣을 수 있게 해주었다. 이후 사용자가 입력할 수 있는 것들을 format 에 맞게 출력하고 Scanner 로 값을 받고 이 값에 따라 다음과 같은 행동을 진행한다.(값들은 if 를 사용해 결정한다.)

Input==1

f.exists()를 이용해 파일이 있는지를 확인, 없을 시 에러를 출력하고 다시 입력으로 돌아간다. 기존에 선언해둔 File f 를 이용, scanner 를 이용해 file 내 내용을 읽고 이 값들을 Tokenizer 를 이용해 ‘,’를 기준으로 끊어낸다. 이 값들을 인자로 Member 객체를 생성해 이를 vector sallist 내에 add 를 통해 넣는다.

Input ==2

이름과 나이를 입력 받고 이들을 vector sallist 내의 Member 들의 인자들과 비교하여 (getage(), getname()이용) name 과 age 가 동시에 같은 경우를 찾는다. 같은 것이 존재할 경우 에러를 출력해주고 없을 경우 연봉을 입력 받아 이 값들을 인자로 Member 객체를 생성해 이를 vector sallist 내에 add 를 통해 넣는다. 이후 sallist 의 인자 전체를 출력해준다.

Input==3

이름과 나이를 입력 받고 이들을 vector sallist 내의 Member 들의 인자들과 비교하여 (getage(), getname()이용) name 과 age 가 동시에 같은 경우를 찾는다. 같은 것이 존재하지 않을 경우 에러를 출력해주고 같은 것이 있을 경우 그때의 vector 요소가 몇 번째인지 기록해 remove 를 이용, 해당 Member 를 제거한다. 이후 sallist 의 인자 전체를 출력해준다.

Input==4

이름과 나이를 입력 받고 이들을 vector sallist 내의 Member 들의 인자들과 비교하여 (getage(), getname()이용) name 과 age 가 동시에 같은 경우를 찾는다. 같은 것이 존재하지 않을 경우 에러를 출력해주고 같은 것이 있을 경우 그때의 vector 요소가 몇 번째인지 기록하고 연봉을 입력받는다. 이후 set 를 이용, 해당 Member 의 연봉을 수정한다. 이후 sallist 의 인자 전체를 출력해준다.

Input==5

Name 을 기준으로 할지 Salary 를 기준으로 할지 입력 받고, 만약 Name 을 기준으로 할 경우 Collections.sort(sallist, new comname());을 이용, 벡터를 이름을 기준으로 내림차순 정렬을 실행한다. Comname 에 내부에서 확인할 수 있듯이, 이름을 기준으로 내림차 순, 이름이 같을 경우 연봉을 기준으로 내림차 순으로 정렬되는 것을 확인할 수 있다.

만약 Salary 를 기준으로 할 경우 Collections sort(saliist, new comint())를 이용, 벡터를 salary 를 기준으로 오름차순 정렬을 실행했다. Comint 의 내부에서 확인할 수 있듯이,

salary 를 기준으로 오름차 순, salary 가 같은 경우 이름을 기준으로 오름차 순으로 정렬되는 것을 확인할 수 있다.

이후 sallist 의 인자 전체를 출력해준다.

Input==6

File.exists()를 이용, file 이 있는지 확인하고 없을 경우 file.createNewFile()를 이용해 파일을 만들어 준다. 이후 기존에 선언해둔 f 를 이용, FileWriter writer=new FileWriter(f)를 선언하고 BufferedWriter wrbuf = new BufferedWriter(writer);를 선언한다. 이후 벡터 내부의 값들을 memstr2 를 통해 return 하고 이 값들을 wrbuf.write 를 통해 txt 파일로 보낸다. 이후 wrbuf.newLine()을 통해 줄을 바꿔준다. 이를 벡터의 끝에 도달할 때 까지 진행한다. 이후 wrbuf 를 닫고 저장 이 성공했음을 알린다. 또한 scanner 를 닫고 프로그램 종료를 console 에 알린다.

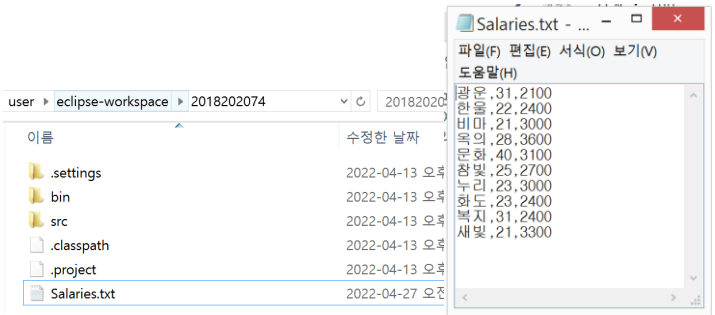
C. Pseudo Code

```
Class Member{
	Private String name;
	Private int age, salary;
	Public Member (String str, int a, int b){this.name=str; this.age=a;this.salary=b;}
	Public String memstr(){return this.name+” / ”+this.age+” / ”+this.salary;}
	Public String memstr(){return this.name+”,”+this.age+”,”+this.salary;}
	Public String getname(){return this.name;}
	Public String getage(){return this.age;}
	Public String getsalary(){return this.salary;}
}
Class comname implements Comparator<Member>{
	Public int compare(Member m1, Member m2){
		If(m1.getname()<m2.getname()){return 1;} //compartTo 이용
		Else If(m1.getname()>m2.getname()){return -1;} //compartTo 이용
		Else{if(m1.getsalary()>m2.getsalary()){return -1;} else{return 1};}
	}
}
Class comint implements Comparator<Member>{
	Public int compare(Member m1, Member m2){
		If(m1.getsalary()>m2.getsalary()){return 1;}
		Else If(m1.getsalary()<m2.getsalary()){return -1;}
		Else{if(m1.getname()<m2.getname()){return -1;} else{return 1};}//compartTo 이용
	}
}
Void main(){
	Vector<Member> sallist = new Vector<Member>();
	File file = new File(“.”);
	String rootPath = System.getProperty(“user.dir”);
```

```
File f = new File(rootPath+"\\Salaries.txt");
Scanner scanner = new Scanner(System.in);
While(true){
        Input format print
        Input = scanner.nextInt();
        If(Input ==1){
                If(file exist){
                                Print "File no exist" to console
                                Break;
                }
                Scanner fileread = new Scanner(f);
                While(fileread.hasNext()){
                StringTokenizer st = new StringTokenizer(str, ",");
                While(next Token exists){make Member Object by Token values; put it in sallist}
                }
                Fileread.close(); print "Load success" in console
        }
        If(Input ==2){
                Take name, age;
                If(sallist have object that have input name & input age)
                  { Print "Already exist" in console;}
                 Else{Take salary; make Member Object by name,age,salary values;put in sallist
                       Print vector all entity by member object
                }
        }
        If(Input ==3){
                Take name, age;
                If(sallist have object that have input name & input age)
                   {remove that object from sallist; Print vector all entity by member object ;}
                 Else{ Print "Not exist" in console;}
        }
        If(Input ==4){
                Take name, age;
                If(sallist have object that have input name & input age)
                   {update that object from sallist; Print vector all entity by member object ;}
                 Else{ Print "Not exist" in console;}
        }
        If(Input==5){
                Take input about option(name, salary)
                If(input==1)Collections.sort(sallist, new comname());
                Else If(input==2)Collections.sort(sallist, new comint());
                Print vector all entity by member object ;
        }
        If(Input == 6){
                If(!file extist) file.createNewFile();
                FileWriter writer = new FileWriter(f);
                BufferedWriter wrbuf = new BufferedWriter(writer);
```

```
                        For(int j=0;j<sallist.size();j++){
                                wrbuf.write(sallist.get(j).memstr2());wrbuf.newLine();
                        }
                        wrbuf.close();
                        print “Save success” in console; break;
                }
        }
        close scanner
        println("Program terminate")
}
```

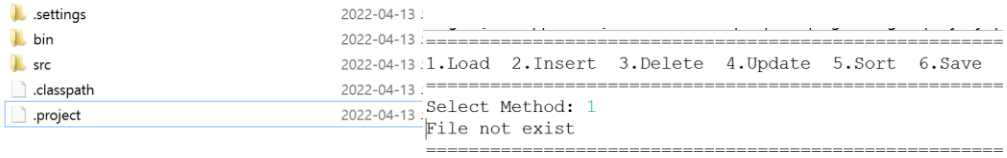
D.Result Screen



해당 위치에 다름과 같은 txt 파일이 있을 경우,

```
============================================================
1.Load  2.Insert  3.Delete  4.Update  5.Sort  6.Save
============================================================
Select Method: 1
Load success
============================================================
```

다음과 같이 출력에 성공한다.



다음과 같이 파일이 없으면 error check 를 해주는 모습니다.

```
============================================================
1.Load  2.Insert  3.Delete  4.Update  5.Sort  6.Save
============================================================
Select Method: 2
이름 입력 : 새빛
나이 입력 : 21
Already exist
```

이미 있는 값을 Insert 를 통해 넣으려고 할 경우 Already exist 라고 에러 체크를 해주는 모습이다.

```
=========================================================
1.Load  2.Insert  3.Delete  4.Update  5.Sort  6.Save
=========================================================
Select Method: 2
이름 입력 : 새빛
나이 입력 : 25
연봉 입력 : 3300
광운 / 31 / 2100
한울 / 22 / 2400
비마 / 21 / 3000
옥의 / 28 / 3600
문화 / 40 / 3100
참빛 / 25 / 2700
누리 / 23 / 3000
화도 / 23 / 2400
복지 / 31 / 2400
새빛 / 21 / 3300
새빛 / 25 / 3300
```

이름이 겹쳐도 나이가 달라 별개의 객체로 판단, Insert 가 성공적으로 이루어져 vector 의 내부 내용들도 잘 출력되고 있다.

```
=====================================================
1.Load  2.Insert  3.Delete  4.Update  5.Sort  6.Save
=====================================================
Select Method: 3
이름 입력 : 새빛
나이 입력 : 28
Not exist
```

Delete 의 경우도 이름이 존재해도 나이가 달라 Delete 가 실패하는 모습이다.

```
=====================================================
1.Load  2.Insert  3.Delete  4.Update  5.Sort  6.Save
=====================================================
Select Method: 3
이름 입력 : 새빛
나이 입력 : 25
광운 / 31 / 2100
한울 / 22 / 2400
비마 / 21 / 3000
옥의 / 28 / 3600
문화 / 40 / 3100
참빛 / 25 / 2700
누리 / 23 / 3000
화도 / 23 / 2400
복지 / 31 / 2400
새빛 / 21 / 3300
```

이름과 나이를 알맞게 입력 시 해당하는 객체가 Vector 내부에서 사라진 걸 확인할 수 있다.

벡터 내의 요소들도 console 에 잘 출력되었다.

```
=========================================================
1.Load  2.Insert  3.Delete  4.Update  5.Sort  6.Save
=========================================================
Select Method: 4
이름 입력 : 누리
나이 입력 : 22
Not exist
```

이름과 나이가 동시에 맞지 않는 경우에는 Not exist 를 출력해 주는 것을 확인 할 수 있다.

```
=========================================================
1.Load  2.Insert  3.Delete  4.Update  5.Sort  6.Save
=========================================================
Select Method: 4
이름 입력 : 누리
나이 입력 : 23
연봉 입력 : 2100
광운 / 31 / 2100
한울 / 22 / 2400
비마 / 21 / 3000
옥의 / 28 / 3600
문화 / 40 / 3100
참빛 / 25 / 2700
누리 / 23 / 2100
화도 / 23 / 2400
복지 / 31 / 2400
새빛 / 21 / 3300
```

이름과 나이가 모두 일치하는 객체를 찾아 연봉을 입력 받고 해당 값을 변경하였다. 벡터에 대한 출력을 통해 벡터 내 값이 알맞게 바뀌었다는 것을 확인할 수 있다.

```
=========================================================
1.Load  2.Insert  3.Delete  4.Update  5.Sort  6.Save
=========================================================
Select Method: 2
이름 입력 : 광운
나이 입력 : 29
연봉 입력 : 2400
광운 / 31 / 2100
한울 / 22 / 2400
비마 / 21 / 3000
옥의 / 28 / 3600
문화 / 40 / 3100
참빛 / 25 / 2700
누리 / 23 / 2100
화도 / 23 / 2400
복지 / 31 / 2400
새빛 / 21 / 3300
광운 / 29 / 2400
```

Sort 를 진행하기 전, 이름이 겹치는 경우를 확인하기 위해 이름이 같은 Object 를 추가하였다.

```
=========================================================
1.Load  2.Insert  3.Delete  4.Update  5.Sort  6.Save
=========================================================
Select Method: 5
1.Name 2.Salary : 1
화도 / 23 / 2400
한울 / 22 / 2400
참빛 / 25 / 2700
옥의 / 28 / 3600
새빛 / 21 / 3300
비마 / 21 / 3000
복지 / 31 / 2400
문화 / 40 / 3100
누리 / 23 / 3000
광운 / 29 / 3000
광운 / 31 / 2100
```

Sort 를 이름기준으로 하자 내림차 순으로 Vector 가 나열되었다.(화,한,참,…,누,광,광 이들은 사전적으로 내림차순이다.) 또한 광운이란 이름을 가진 객체들은 연봉이 더 큰 쪽이 위로 오도록 나열되었다.

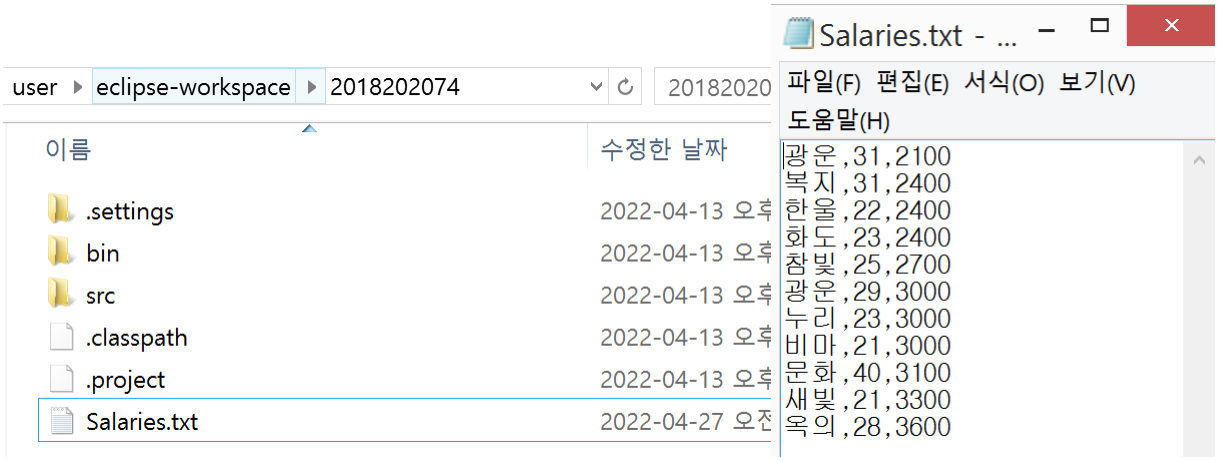
```
=========================================================
1.Load  2.Insert  3.Delete  4.Update  5.Sort  6.Save
=========================================================
Select Method: 5
1.Name 2.Salary : 2
광운 / 31 / 2100
복지 / 31 / 2400
한울 / 22 / 2400
화도 / 23 / 2400
참빛 / 25 / 2700
광운 / 29 / 3000
누리 / 23 / 3000
비마 / 21 / 3000
문화 / 40 / 3100
새빛 / 21 / 3300
옥의 / 28 / 3600
```

Sort 를 연봉기준으로 하자 오름차 순으로 Vector 가 나열되었다. 또한 연봉이 같은 경우에는 이름이 사전적 순서대로 오름차 순으로 되어있다는 것을 확인 할 수 있다.

(2400 연봉을 가지는 객체들의 이름이 복지, 한울, 화도 순으로 나열되었으며 이는 사전적으로 오름차 순이다.)

```
=========================================================
1.Load  2.Insert  3.Delete  4.Update  5.Sort  6.Save
=========================================================
Select Method: 6
Save success
Program terminate
```

새로 파일을 Load 한 후 Salary 를 기준으로 Sort 한 후 6 을 입력하여 Save 를 실행해주었다.



파일이 없는 경우, 해당 위치에 다음과 같은 내용을 지닌 Salaries.txt.파일이 생성되었고 있는 경우에는 내부 파일이 위와 같은 내용을 가지도록 수정되었다.

E.고찰

위의 경우 main 함수가 상당히 길어져 있음을 알 수 있다. 덕분에 사람이 다시 읽고 쓰기에 보기 좋지 않다. 각 Input 값에 따라 함수를 따로 지정해 main 함수에서 부르는 형식이었다면 사람이 읽기 더 편했을 것이다. 또한 제시된 조건 외에도 예외처리를 할 수 있는 부분이 많았다.(ex)위 문제에서는 1-6 사이의 input 이 들어온다고 가정하고 있지만 실제로 문자열 등을 입력하게 될 경우 에러가 발생한다. 실제로 사용되는 프로그램의 경우 이러한 점들을 잡아주는 것이 중요할 것이다. 추가적으로 컬렉션을 사용하라는 조건에 의해 Vector 를 사용하였는데, 위의 배열의 경우 Key 값을 2 개를 사용하여 Class 자체에 인자를 통해 접근할 필요가 있어 다음과 같이 진행하였고, 조건이 달랐다면 Arraylist 나 HashMap(특히 Key 가 한 개였다면 이를 사용했을 것이다.)도 사용하는 것이 용이했을지 모르겠다.