2019학년도 2학기

JAVA 프로그래밍 II

514770-2
2019년 가을학기
9/11/2019
박경신

과제 Lab2_1

1. 본 과제에서 작성하는 코드는 Lab2_1.java 파일에 저장한다.
2. 재귀호출(recursive call)을 사용하여, 다음 메소드(method) 프로그램을 작성하라. 이 프로그램은 사용자 입력을 받아서 계산하여 출력한다.
   1. 양의 정수 n의 n! (팩토리얼) 메소드를 작성하라.
   2. 양의 정수 n의 f(n) (피보나치 수열) 메소드를 작성하라.
   3. 양의 정수 b, n의 b^n (파워) 메소드를 작성하라.
   4. 양의 정수 a, b의 최대공약수 메소드를 작성하라.
   5. **본인이 원하는 재귀호출 메소드를 추가로 작성하라.**

과제 Lab2_2

1. Lab2_2 에서는 체감추위지수 Wind Chill Temperature (WCT) Index를 계산하는 프로그램을 작성한다.
   http://www.kma.go.kr/HELP/basic/help_01_07.jsp
2. enum WindChillTemperatureIndex { DANGER, WARNING, CAUTION, AWARE }
   - WindChillTemperatureIndex getIndex() 체감온도 지수를 반환한다.
3. WindChillTemperatureIndexCalculator 클래스에서는
   - double calculate(double T, double V) 체감온도 값을 계산한다.

체감온도값 = 13.12 + 0.6215*T - 11.37 * V^0.16 + 0.3965 * V^0.16 * T [T: 기온(섭씨), V : 풍속(km/h)]

   - 체감온도 산출표와 사용자 입력을 받아서 체감온도를 출력하거나 등 routine을 추가한다.

과제 Lab2_3

1. Lab2_3 에서는 열지수 Heat Index를 계산하는 프로그램을 작성한다.
   http://www.kma.go.kr/HELP/basic/help_01_04.jsp
2. enum HeatIndex { VERY_HIGH, HIGH, USUAL, LOW }
   - HeatIndex getIndex() 체감온도 지수를 반환한다.
3. HeatIndexCalculator 클래스에서는
   - double calculate(double F, double R) 열지수 값을 계산한다.

열지수값 = -42.379 + (2.04901523*F) + (10.14333127*R) - (0.22475541*F*R) - (0.00683770*F*F) - (0.05481717*R*R) + (0.00122874*F*F*R) + (0.00085282*F*R*R) - (0.00000199*F*F*R*R) [F: 화씨온도, R: 상대습도]

   - 열지수 산출표와 사용자 입력을 받아서 열지수를 출력하거나 등 routine을 추가한다.

과제 Lab2_4

1. Lab2_4 에서는 부패지수 Decomposition Index를 계산하는 프로그램을 작성한다. http://www.kma.go.kr/HELP/basic/help_01_03_01.jsp
2. enum DecompositionIndex { HIGH, NORMAL, LOW }
   - DecompositionIndex getIndex() 부패지수를 반환한다.
3. DecompositionIndexCalculator 클래스에서는
   - double calculate(double RH, double T) 부패지수 값을 계산한다.

   부패지수값 = ((RH - 65)/14) * (1.054*T) [RH: 상대습도 (%), T: 기온 (섭씨)]
   - 부패지수 산출표와 사용자 입력을 받아서 부패지수를 출력하거나 등 routine을 추가한다.

과제 Lab2_5

1. Lab2_5 에서는 본인이 원하는 다른 각종 지수 (생활기상/보건기상 등등) 프로그램을 작성한다.

재귀호출 (Recursive call)

https://introcs.cs.princeton.edu/java/23recursion/

- 함수에서 자기 자신을 다시 부르는 것을 재귀호출 (recursive call)이라고 함
- 함수 내부에서 사용되는 지역 변수의 값들은 자기 자신을 호출하기 전의 값들을 호출 후에도 그대로 보존함 (자기 자신을 다시 부르더라도 새로운 지역 변수들이 생성되는 것을 생각하면 됨)
- 재귀호출을 빠져나갈 수 있도록 검사하고 종료하는 부분이 반드시 존재해야 함 (재귀호출 탈출 조건)
- 재귀호출의 사용 예: 팩토리얼 (factorial), 최대공약수, 피보나치 수열, 등

재귀호출

- 팩토리얼을 구현하는 함수 작성
- n! = n x (n – 1) x (n – 2) x … x 1 = n x (n – 1)! (where n >= 1)
- 알고리즘
  - 만약 n이 0보다 작거나 같으면 값 1을 반환
  - 만약 n이 0보다 크다면 n * (n – 1)!을 반환

$$n! = \begin{cases} 1 & if n = 0 \\ n(n-1)! & if\ n > 0 \end{cases}$$

```
static int factorial(int n){
  if (n == 0)
    return 1;
  else
    return(n * factorial(n-1));
}

factorial(1);  // 1
factorial(5);  // 5! = 5 * 4 * 3 * 2 * 1 = 120
```

재귀호출

- 피보나치 수열 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233
- 알고리즘
  - 1항과 2항은 1
  - 3항 이후부터의 n항은 (n – 1)항 + (n – 2)항

$$F(n) = F(n-1) + F(n-2)\ for\ n \geq 2, with\ F(0) = 0\ and\ F(1) = 1$$

```
static long fibonacci(int n) {
    if (n == 0) return 0;
    if (n == 1) return 1;              // 재귀호출 탈출 조건
    return fibonacci(n – 1) + fibonacci(n – 2);
}
fibonacci(1)                // 1
fibonacci(2)                // 1
fibonacci(3)                // 2
fibonacci(6)                // 8
```

재귀호출

- power 함수 작성
- 알고리즘
  - 만약 n이 0이면, 값 1을 반환
  - 만약 n이 0이 아니라면, b * power(b, n – 1)을 반환

$$b^n = \underbrace{b \times \cdots \times b}_{n}.$$

```
static int power(int b, int n){
  if (n == 0)
    return 1;
  else
    return(b * power(b, n-1));
}

power(3, 2);  // 3^2 = 3 * 3 = 9
power(5, 5);  // 5^5 = 5 * 5 * 5 * 5 * 5 = 3125
```

$$b^n = \begin{cases} 1 & if n = 0 \\ b * b^{(n-1)} & if\ n > 0 \end{cases}$$

재귀호출

- 최대공약수(Greatest Common Divisor)
- 알고리즘
  - 만약 a>b, gcd(a,b) = gcd(b, a%b)

$$\gcd(a,b) = \begin{cases} a & if b = 0 \\ gcd(b, a\%b) & if\ b > 0 \end{cases}$$

```
static int gcd(int a, int b) {
    if (b == 0)
        return a;              // 재귀호출 탈출 조건
    else
        return gcd(b, a%b);
}
gcd(12, 4);              // 4
gcd(12, 18);             // 6
```

과제 제출

- Lab2_1 ~ Lab2_5와 보고서를 전체적으로 묶어서 e-learning에 과제 제출