3
산술연산과 변수

학습 목표

- 기본 명령문들 (변수 선언, 사용, 저장문) 이해
- 다양한 타입 이해
  - 정수, 실수, 논리값, 문자, 문자열
- 타입 간 변환 이해

3.1 동전 계산 프로그램

- 미국의 동전
  - quarter = 25-cent
  - dimes = 10-cent
  - nickels = 5-cent
  - pennies = 1-cent

- 한국에 사는 것이 다행이군! 계산하기 복잡해.
  - 9 quarters, 2 dimes, no nickels, and 6 pennies

    (9×25)+(2×10)+(0×5)+(6×1) = 251 cents = $2.51

- 어쨌든, 이것을 계산하는 프로그램을 작성해 보자.

정수 연산

- 정수는 수학에서 쓰듯이 쓰면 된다.
  - 2, 100, -10
- 정수 연산은
  - 더하기: +
  - 빼기: -
  - 곱하기: *
  - 나누기: /
  - 괄호도 사용 가능, 예, (1+2)*3

동전 계산 프로그램: Coins.java

```
/** Coins - 9 quarters, 2 dimes, no nickels, and 6
pennies 를 가지고 있을 때 몇 센트를 가지고 있는지 계산해 주는 프로그램
*/

public class Coins
{  public static void main(String[] args)
   {  System.out.println("For 9 quarters, 2 dimes,
                          no nickels, and 6 pennies,");
      System.out.print("the total is ");
      System.out.println( (9 * 25) + (2 * 10)
                        + (0 * 5) + (6 * 1) );
   }
}
```

*이 곱하기

```
For 9 quarters, 2 dimes, no nickels, and 6 pennies,
the total is 251
```

3.2 penny 동전 개수가 바뀌면?

```
/** Coins - 9 quarters, 2 dimes, no nickels, and 6
pennies 를 가지고 있을 때 몇 센트를 가지고 있는지 계산해 주는 프로그램
*/

public class Coins
{  public static void main(String[] args)
   {  System.out.println("For 9 quarters, 2 dimes,
                          no nickels, and 6 pennies,");
      System.out.print("the total is ");
      System.out.println( (9 * 25) + (2 * 10)
                        + (0 * 5) + (6 * 1) );
   }
}
```

두 곳을 수정해 주어야 한다 --;

```
For 9 quarters, 2 dimes, no nickels, and 6 pennies,
the total is 251
```

변수 (Variable)

- 값을 저장할 수 있는 저장소

- 이름 만드는 규칙
  - 영문자, 숫자, _, $를 혼용해서 사용 가능, 단, 숫자로 시작 금지
  - 예, `quarters`, `QUArTers`, `Q123`, `my_quarter`
  - Java 키워드는 사용 금지: `public`, `class` 등

- 변수 선언과 초기화
  - `int quarters = 5;`
- 변수 사용
  - `System.out.println(quarters * 25);`

변수 사용해서 프로그램 작성하기

```
/** CoinsVariables - 네 개의 변수로 quarter, dimes, nickels, pennies를
몇 개씩 가지고 있는지 정의해서 사용 */

public class CoinsVariables
{ public static void main(String[] args)
  { int quarters = 9;
    int dimes = 2;
    int nickels = 0;
    int pennies = 6;
    System.out.println("For these quantities of coins:");
    System.out.print("Quarters = "); System.out.println(quarters);
    System.out.print("Dimes = "); System.out.println(dimes);
    System.out.print("Nickels = "); System.out.println(nickels);
    System.out.print("Pennies = "); System.out.println(pennies);
    System.out.print("The total is ");
    System.out.println((quarters*25)+(dimes*10)
                       +(nickels*5)+(pennies*1) );
  }
}
```

결과

```
For these quantities of coins:
Quarters = 9
Dimes = 2
Nickels = 0
Pennies = 6
The total is 251
```

여러 개 출력하기

- 출력할 때 마다 `System.out.print` 또는 `println`을 호출해야 하니까 불편하다.
- `System.out.print(x); System.out.print(y);` 는 `System.out.print(x+y);` 로 짧게 표기 가능하다.
- `System.out.print(x); System.out.println(y);` 는 `System.out.println(x+y);` 로 짧게 표기 가능하다.
- 두 개 이상도 가능하다.
- 정수 +와 헷갈릴 수 있으므로 유의하라.

출력 짧게

```
/** CoinsVariables - 네 개의 변수로 quarter, dimes, nickels, pennies를
몇 개씩 가지고 있는지 정의해서 사용 */

public class CoinsVariables
{ public static void main(String[] args)
  { int quarters = 9;
    int dimes = 2;
    int nickels = 0;
    int pennies = 6;
    System.out.println("For these quantities of coins:");
    System.out.println("Quarters = " + quarters);
    System.out.println("Dimes = "    + dimes);
    System.out.println("Nickels = "  + nickels);
    System.out.println("Pennies = "  + pennies);
    System.out.println("The total is " +
                       ((quarters*25)+(dimes*10)
                        +(nickels*5)+(pennies*1)));
  }
}
```

달러로 표기해 줘

- 251 센트가 아니라 $2.51로 표기해 줘!
- 힌트
  - 251 / 100 = 2
  - 251 % 100 = 51
    - %는 수학에서 mod, 즉, 나머지를 주는 연산자이다.
- 251을 두 번 사용하네.
  - 변수를 쓰자.

동전 계산 결과를 달러로 표기하는 프로그램

```
/** TotalVariablesDollar - quarters, dimes, nickels, pennies 개수
가 주어지면 얼마인지 달러로 계산하는 프로그램 */

public class TotalVariablesDollar
{ public static void main(String[] args)
  {
    ……
    System.out.print("The total is ");
    int total = (quarters * 25) + (dimes * 10)
              + (nickels * 5) + (pennies * 1);
    System.out.print("The total is $");
    System.out.print(total / 100);
    System.out.print(".");
    System.out.println(total % 100);
  }
}
```

변수 선언과 초기화

- 변수가 선언되고 초기화된다는 것은, 메모리에 "저장소 셀(cell)"이 생기고 초기화 값이 그 셀에 저장된다는 것이다.

```
main
{    int quarters = 9
     int dimes    = 2
     int nickels  = 0
     int pennies  = 6
   ...
}
```

저장 또는 대입 (Assignment)

- 변수 저장소 셀에 있는 값은 바꿀 수 있다.

```
int money = 100;
System.out.println(money);
money = 0;
System.out.println(money);
```

```
int money = 100;
System.out.println(money);
money = money + 50;
System.out.println(money);
```

잔돈 계산: 문제

- 입력: 3 달러 46 센트를 줘 봐! 동전으로.
- 출력:
  - quarters = 13
  - dimes = 2
  - nickels = 0
  - pennies = 1

- 어떻게 계산하지?
  - 3.46 - (13×0.25) = 0.21
  - 0.21 - (2×0.10) = 0.01
  - 0.01 - (0×0.05) = 0.01
  - 0.01 – (1×0.01) = 0.00

잔돈 계산: 알고리즘

- 시작 금액을 money라고 하자.
- money보다 작게 되는 최대 quarter 수를 계산한다. 해당 금액을 money에서 빼고 quarter 수를 출력한다.
- money보다 작게 되는 최대 dime 수를 계산한다. 해당 금액을 money에서 빼고 dime 수를 출력한다.
- money보다 작게 되는 최대 nickel 수를 계산한다. 해당 금액을 money에서 빼고 nickel 수를 출력한다.
- 남은 money가 penny의 수이다.

잔돈 계산: 프로그램

```
/** MakeChange- dollars와 cents가 주어지면 이에 해당하는 동전 수를 계산한
다. */

public class MakeChange
{ public static void main(String[] args)
  { int dollars = 3;
    int cents = 46;
    int money = (dollars * 100) + cents;
    System.out.println("quarters = " + (money / 25));
    money = money % 25;
    System.out.println("dimes = " + (money / 10));
    money = money % 10;
    System.out.println("nickels = " + (money / 5));
    money = money % 5;
    System.out.println("pennies = " + money);
  }
}
```

잔돈 계산: 실행 과정

```
main
   { int dollars =  3
     int cents   =  46
     int money   =  346
     System.out.println("quarters = " + (money / 25));
     money = money % 25;
     System.out.println("dimes = " + (money / 10));
     money = money % 10;
     … }
```

```
main
   { int dollars =  3
     int cents   =  46
     int money   =  21
     System.out.println("quarters = " + (money / 25));
     money = money % 25;
     System.out.println("dimes = " + (money / 10));
     money = money % 10;
     … }
```

`money = money % 25` 의 실행

- money 변수의 셀을 찾는다.
- money % 25 를 계산한다.
  - money 변수의 값을 읽는다.
  - 25로 나눈 나머지를 구한다.
- 계산 결과를 money 변수의 셀에 저장한다.
  - 기존에 있던 값은 사라진다.

선언과 초기화는 분리할 수 있다

- 선언
  - `int dollars;`
  - dollars를 위한 셀만 생성한다.

- 저장 (초기화)
  - `dollars = 3;`
  - 3을 dollars 셀에 저장한다.

3.3 유리수 계산: Doubles

- 온도 변환 프로그램을 만들자.

- 계산 공식: f = (9/5)c + 32
  - c 는 섭씨 온도
  - f 는 화씨 온도
  - c = 22 이면 f = 71.6

- 유리수를 Java로?
  - 유리수는 Java에서 기본으로 지원하지 않는다.
  - 대신 실수 double을 사용할 수 있다.
  - 실수는 수학에서 쓰듯이 1.2, -2.1, 0.33 등으로 표기한다.
  - 실수연산자는 +, -, *, / 를 정수와 같이 사용한다.

온도 변환 프로그램

```
/** CelsiusToFahrenheit0 - 섭씨를 화씨로 변환 */

public class CelsiusToFahrenheit0
{ public static void main(String[] args)
  { int c = 22;       // the degrees Celsisus
    double f = ((9.0/5.0) * c) + 32;
    System.out.println("For Celsius degrees "+ c + ",");
    System.out.println("Degrees Fahrenheit = " + f);
  }
}
```

묵시적 타입변환

```
For Celsius degrees 22,
Degrees Fahrenheit = 71.6
```

정수와 실수의 타입 변환 (Type Cast)

- 이게 될까?
  - `int i = 1;`
  - `double d = i;` 묵시적 타입 변환 (implicit type casting)

- 이게 될까?
  - `double d = 1.5;`
  - `int i = d;`

- 위에 것은 되고 아래 것은 안된다. 이렇게 써 주어야 한다.
  - `double d = 1.5;`
  - `int i = (int)d;` 명시적 타입 변환 (explicit type casting)

원의 면적을 정수로 구하기

```
/** Geometry - 원의 면적을 정수로 구하기 */

public class Geometry
{ public static void main(String[] args)
  { int radius = 7;
    int area;
    area = (int)(3.14 * radius * radius);
    System.out.println(area);
  }
}
```

명시적 타입변환

```
153
```

3.4 논리값 (Boolean Value)

- 참(true)/거짓(false)을 나타내는 값
  - `boolean b = false;`
- 비교 연산자: 정수, 실수 모두 비교 가능하다.
  - `> < <= >= == !=`
    - 주의, 수학기호에서 비교는 = 이지만 Java에서는 저장문과 혼동을 피하기 위해 ==
    - ≦, ≧, ≠는 유사한 기호로 표기
- 논리 연산자
  - ∧, ∨, ¬는 `&&`, `||`, `!`으로
- 예, `x<y && !(y >= 20)`

3.5 연산자 우선순위 (Operator Precedence)

- 1*2+3
  - (1*2)+3 일까, 아니면 1*(2+3) 일까?
  - 혼동을 막기 위해 우선순위를 부여하였다.

단항 연산자 - !	높다
곱하기 * 나누기 / 나머지 %	
더하기 + 빼기 -	
비교 < <= > >=	
비교 == !=	
논리 && ||	낮다

연산자 결합순서 (Associativity)

- 1-2-3
  - (1-2)-3 일까, 아니면 1-(2-3) 일까?
  - 혼동을 막기 위해 결합순서를 부여하였다.

- 일반적으로, 단항연산자는 우결합성 이항연산자는 좌결합성
  - 단항연산자 -: `----4` = -(-(-(-4)))
  - 이항연산자 -: `1-2-3` = (1-2)-3

3.6 문자열 (String)

- 문자들의 나열
  - 문자열은 정수, 실수, 논리값과는 달리 “객체”이다.
  - java.lang 에 class String 정의되어 있음
    - `String name = “Gildong Hong";`
- 문자열 붙이기 (concatenation)
  - `System.out.println("My name is " + name);`
  - `System.out.println("My name is R2D" + (5-3));`

문자열 클래스의 메소드

- `S1.equals(S2), S1.compareTo(S2)`
- `S.length()`
- `S.charAt(E), S1.indexOf(S2,i)`
- `S.substring(E1,E2)`
- `S.toUpperCase(), S.toLowerCase()`
- `S.trim`

- 자세한 내용은 API 문서 참조

문자열 연산

```
public class StringDemo {
    public static void main(String[] args) {
        String slogan_eng = "Freedom is not free";
        System.out.println(slogan_eng);
        System.out.println(slogan_eng.toUpperCase());
        System.out.println(slogan_eng.charAt(5));
        System.out.println(slogan_eng.substring(4,6));
        System.out.println(slogan_eng.indexOf('o'));
        System.out.println(slogan_eng.indexOf('o',6));
        System.out.println("a".compareTo("d"));
    }
}
```

```
Freedom is not free
FREEDOM IS NOT FREE
o
do
5
12
-3
```

문자 (Characters)

- 글자 하나
  - 일반 문자: `'a' 'b' '0' '&'`
  - 특수 문자: `'\b' '\t' '\n' '\r' '\"' '\'' '\\'`

- 문자는 정수로 변환 가능하다.
  - 각 문자는 컴퓨터에서 특정 숫자에 대응된다.
  - Java의 경우 유니코드(Unicode) 방법에 따라 대응시킨다.
  - 정수와 상호 변환이 가능하다.
  - `(char)('a' + 1)`?

3.7 타입 이야기

- 값 들의 종류를 타입 (data type) 이라고 한다.

- Java는 두 가지 다른 카테고리의 타입이 있다.
  - 기본 타입 (primitive type): `int`, `double`, `boolean`, `char`
  - 참조 타입 (reference type) 또는 객체 타입 (object type): `String`, `GregorianCalendar`

- Java는 엄격하게 타입을 강요한다. 왜?
  - 이상한 연산 불가: `1 * 2.3` (O), `"abc" * 2.3` (X)
  - 연산 결과 예측 가능: `1 * 2.3` 는 뭔지 몰라도 double 값이지
  - 변수 셀의 할당/이용 용이: `double`은 2셀에 저장할 수 있고 `double` 값만 가지지
  - 프로그램 오류 쉽게 검출: 프로그램이 잘못 작성되면 타입부터 틀릴 가능성 높음

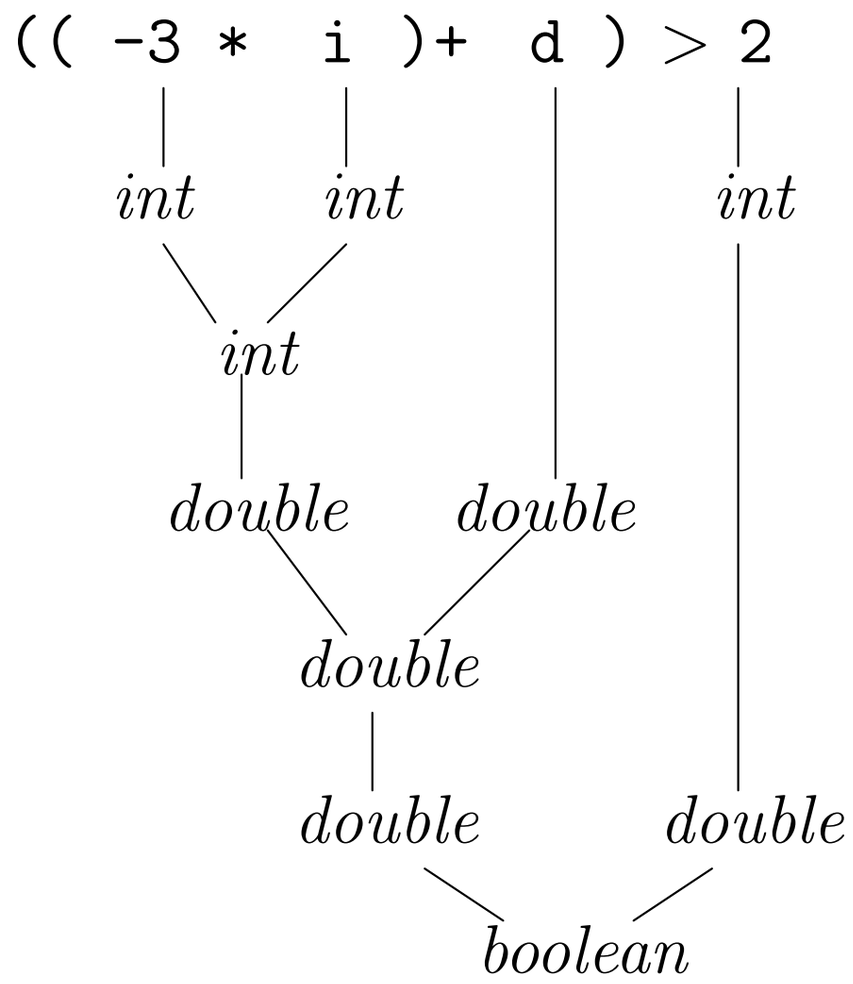
타입 오류

```
boolean b = true;
System.out.println(b * 5);      // data type error
int i = 3 * 2.1;                // data type error
int x;
x = "abc";                      // data type error

GregorianCalendar c = new GregorianCalendar();
System.out.println(c.getTime());
c.println("oops");               // data type error
```

타입 오류가 있는 프로그램은 대부분의 경우 컴파일러가 잡아 낸다.

타입 검사


((-3 * i)+ d) > 2
int
int
int
int
double
double
double
double
double
boolean

3.8 다시 온도 변환 프로그램

- 섭씨를 화씨로 바꾸는 프로그램을 작성하자.
- 이번에는 사용자에게 입력을 받아서 해보자.

- 문제: 입력을 어떻게 받지?

3.8 명령어 줄에서 인수 넘기기

- 프로그램에 입력을 주는 가장 간단한 방법
- 어떻게?
  - 명령어 줄에서: `java  CelsiusToFahrenheit  20`
  - Eclipse에서: `Run configuration > Arguments > Program arguments`
    혹은 `Run configuration > Arguments > Program arguments` 아래 `Variables… > string_prompt` 클릭. Run 실행시 팝업창에 명령행 인자 입력가능.
- 프로그램이 시작되면 `args[0]`, `args[1]`, ..., `args[n]`에 인수가 저장되어 있다.

예제

```
/** LengthOfName - 입력의 길이를 출력한다. */

public class LengthOfName
{ public static void main(String[] args)
  { String name = args[0]; // args[0]에 인수가 저장되어 있다.
    int length = name.length();
    System.out.println("The name, " + name +
                       ", has length " + length);
  }
}
```

```
> java LengthOfName "HONG Gildong"
The name, HONG Gildong, has length 12
```

```
> java LengthOfName HONG Gildong
The name, HONG, has length 4                    args[1] = "Gildong"
```

문자열 <-> 정수, 실수

- 인수는 문자열이다.
  - 20을 넘겨도 정수 20이 아니고 문자열 "20"이다.

- 문자열 "20"을 정수로 바꾸는 방법
  - `int c = Integer.parseInt("20");`

- 실수도 같은 방법으로
  - `double c = Double.parseDouble("3.14");`

- `java.lang` 패키지에 `Integer`와 `Double` 클래스가 있다.

숫자 이쁘게 찍기

- 온도를 계산하고 나면 복잡한 실수가 나올 것이다.

```
System.out.println(100.0/3.0);
```

```
33.333333333333336
```

- 이쁘게 찍는, 즉, 33.33이라고 찍는 방법이 없을까?
  - java.text.DecimalFormat을 사용해!

DecimalFormat

- DecimalFormat 객체 만들기
  - new DecimalFormat(<패턴>)
  - <패턴>은 어떤 모양으로 할 지 적어 놓은 문자열
    - 예, “0.00” = 최소 왼쪽에 한자리, 오른쪽은 두자리

- 이제 이 객체에 format 메소드를 통해 값을 보내면 문자열을 준다.
  - `DecimalFormat f = new DecimalFormat(“0.00”);`
  - `String s = f.format(100.0/3.0);`

온도 변환 프로그램 완성

```
import java.text.*

/** CelsiusToFahrenheit1 는 프로그램 인수로 섭씨온도를 받아 화씨온도로 바꾸
어 출력해 준다 */

public class CelsiusToFahrenheit1 {
 public static void main(String[] args)
 {
  int c = Integer.parseInt(args[0]);
  // args[0]가 프로그램 인수
  double f = ((9.0/5.0)*c) + 32;
  System.out.println("For Celsius degrees " + c + ",");
  DecimalFormat formatter = new DecimalFormat(“0.0”);
  System.out.println("Degrees Fahrenheit = " +
                     formatter.format(f));
 }
}
```

3.9 오류 오류 오류

- 문법 오류
  - 모양이 이상한 프로그램
    - 예, `System.out.println ( (1+2(*3 );`
  - 컴파일러가 잡아 준다.

- 타입 오류
  - 타입이 맞지 않는 프로그램
    - 예, `System.out.println(3 + true);`
  - 컴파일러가 잡아 준다.
  - 변수 선언이 맞지 않는 경우도 타입 검사 중에 잡아 준다.
    - 예, `System.out.println(a); // a 선언 없이 사용`
    - 예, `int a=1; double a=2.5; // 변수 중복 선언`

```
System.out.println(a);
```

실행시간 오류 (Run-Time Error)

- 일부 오류는 실행해 보지 않고는 찾기 어려워 실행시간에 발생한다.
  - 예, 0으로 나누기

```
i/x // x=0
```

  - 예, 변환 오류

```
Integer.parseInt(“abc”)
```

- Java는 실행시간 오류가 발생하면 예외(exception)을 발생시키면서 발생 경로를 알려준다.

```
java.lang.NumberFormatException: abc
        at java.lang.Integer.parseInt(Integer.java)
        at java.lang.Integer.<init>(Integer.java)
        at Test.main(Test.java:4)
```

실행시간에도 여전히 잡히지 않는 오류가 있다

- 논리적 오류를 제외하고서라도 문법적 실수가 문법적으로 문제가 없어서 그냥 실행되는 경우도 있다.

```
int x=3;
int y=7;
System.out.println(x=y);
```

요약

- 변수 선언, 사용, 저장문
- 정수, 실수, 논리값, 문자, 문자열
- 타입, 타입 자동 변환