### 2.1 Functions

**Functions (Static Methods)**

- Takes zero or more input arguments.
- Returns one output value.
- Side effects (e.g., output to standard draw).

**Applications.**
- Scientists use mathematical functions to calculate formulas.
- Programmers use functions to build modular programs.
- You use functions for both.

**Examples.**
- Our I/O libraries: `StdIn.readInt()`, `StdDraw.line()`, `StdAudio.play()`.
- User-defined functions: `main()`.

### Anatomy of a Java Function

**Java functions.** Easy to write your own.

```
2.0  \rightarrow  f(x) = \sqrt{x}  \rightarrow  1.414213...
```

```
public static double sqrt(double c)
{
    if (c < 0) return Double.NaN;
    double err = 1e-15;
    double t = c;
    while (Math.abs(c - t*t) > err * t)
        t = (c/t + t) / 2.0;
    return t;
}
```
Flow of Control

**Key point.** Functions provide a new way to control the flow of execution.

What happens when a function is called:
- Control transfers to the function code.
- Argument variables are assigned the values given in the call.
- Function code is executed.
- Return value is assigned in place of the function name in calling code.
- Control transfers back to the calling code.

Note. This is known as "pass by value."

Scope

**Scope (of a name).** The code that can refer to that name.

Ex. A variable's scope is code following the declaration in the block.

Best practice: declare variables to limit their scope.

Function Challenge 1a

Q. What happens when you compile and run the following code?

```java
public class Cubes1 {
    public static int cube(int i) {
        int j = i * i * i;
        return j;
    }
    public static void main(String[] args) {
        int N = Integer.parseInt(args[0]);
        for (int i = 1; i <= N; i++)
            StdOut.println(i + " " + cube(i));
    }
}
```

% javac Cubes1.java % java Cubes1 6

1 1 2 8 3 27 4 64 5 125 6 216

Function Challenge 1b

Q. What happens when you compile and run the following code?

```java
public class Cubes2 {
    public static int cube(int i) {
        int j = i * i * i;
        return j;
    }
    public static void main(String[] args) {
        int N = Integer.parseInt(args[0]);
        for (int i = 1; i <= N; i++)
            StdOut.println(i + " " + cube(i));
    }
}
```

Function Challenge 1c

Q. What happens when you compile and run the following code?

```java
public class Cubes3 {
    public static int cube(int i) {
        i = i * i * i;
    }
    public static void main(String[] args) {
        int N = Integer.parseInt(args[0]);
        for (int i = 1; i <= N; i++)
            StdOut.println(i + " " + cube(i));
    }
}
```

Function Challenge 1d

Q. What happens when you compile and run the following code?

```java
public class Cubes4 {
    public static int cube(int i) {
        i = i * i * i;
        return i;
    }
    public static void main(String[] args) {
        int N = Integer.parseInt(args[0]);
        for (int i = 1; i <= N; i++)
            StdOut.println(i + " " + cube(i));
    }
}
```
**Function Challenge 1e**

Q. What happens when you compile and run the following code?

```
public class Cubes5 {
    public static int cube(int x) {
        return x * x * x;
    }

    public static void main(String[] args) {
        int N = Integer.parseInt(args[0]);
        for (int i = 1; i <= N; i++)
            StdOut.println(i + " * " + cube(i));
    }
}
```

---

**Gaussian Distribution**

*Standard Gaussian distribution.*

- "Bell curve."
- Basis of most statistical analysis in social and physical sciences.

**Ex.** 2000 SAT scores follow a Gaussian distribution with mean $\mu = 1019$, stddev $\sigma = 209$.

**Gaussian Cumulative Distribution Function**

Goal. Compute Gaussian cdf $\Phi(z)$.

Challenge. No "closed form" expression and not in Java library.

**Bottom line.** 1,000 years of mathematical formulas at your fingertips.

---

**Java Function for $\phi(x)$**

Mathematical functions. Use built-in functions when possible; build your own when not available.

```
public class Gaussian {
    public static double phi(double x) {
        return Math.exp(-(x * x) / 2) / Math.sqrt(2 * Math.PI);
    }

    public static double phi(double x, double mu, double sigma) {
        return phi(x - mu) / sigma;
    }
}
```

Overloading. Functions with different signatures are different.

Multiple arguments. Functions can take any number of arguments.

Calling other functions. Functions can call other functions.

---

**Java function for $\Phi(z)$**

```
public class Gaussian {
    public static double Phi(double x) {
        if (x < -80) return 0.0;
        if (x > 80) return 1.0;
        double sum = 0.0; term = x;
        for (int i = 1; sum + term != sum; i *= 2) {
            term = term * x / i;
            sum += term;
        }
        return 0.5 + sum * phi(x);
    }

    public static double Phi(double x, double mu, double sigma) {
        return Phi(x - mu) / sigma;
    }
}
```

Accurate with absolute error less than $1 \times 10^{-16}$.
SAT Scores

Q. NCAA requires at least 820 for Division I athletes. What fraction of test takers in 2000 do not qualify?

A. \( \Phi(820, 1019, 209) \approx 0.17051 \). [approximately 17%]

Gaussian Distribution

Q. Why relevant in mathematics?
A. Central limit theorem: under very general conditions, average of a set of random variables tends to the Gaussian distribution.

Q. Why relevant in the sciences?
A. Models a wide range of natural phenomena and random processes.
   - Weights of humans, heights of trees in a forest.
   - SAT scores, investment returns.

Caveat:

“Everybody believes in the exponential law of errors: the experimenters, because they think it can be proved by mathematics; and the mathematicians, because they believe it has been established by observation.”
— M. Lippman in a letter to H. Poincaré

Building Functions

Functions enable you to build a new layer of abstraction.
   - Take you beyond pre-packaged libraries.
   - You build the tools you need: Gaussian.ph(), ...

Process.
   - Step 1: identify a useful feature.
   - Step 2: implement it.
   - Step 3: use it.
   - Step 3': re-use it in any of your programs.

Function Examples

- absolute value of an int value
  ```java
  public static int abs(int x) {
      if (x < 0) return -x;
      else return x;
  }
  ```

- absolute value of a double value
  ```java
  public static double abs(double x) {
      if (x < 0.0) return -x;
      else return x;
  }
  ```

- primality test
  ```java
  public static boolean isPrime(int N) {
      if (N < 2) return false;
      for (int i = 2; i <= N/2; i++)
          if (N % i == 0) return false;
      return true;
  }
  ```

- hypotenuse of a right triangle
  ```java
  public static double hypotenuse(double a, double b) {
      return Math.sqrt(a*a + b*b);
  }
  ```

Extra Slides