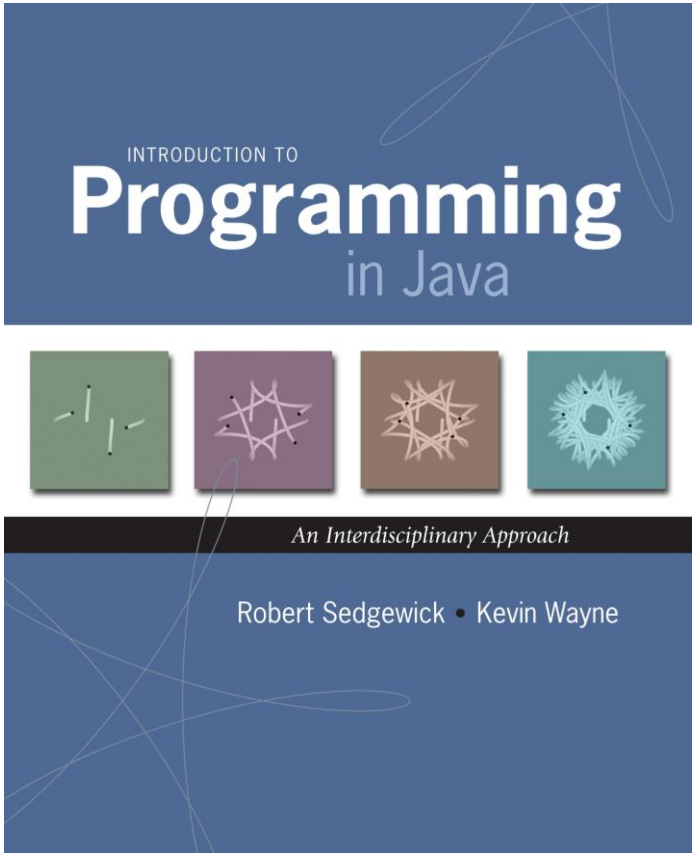
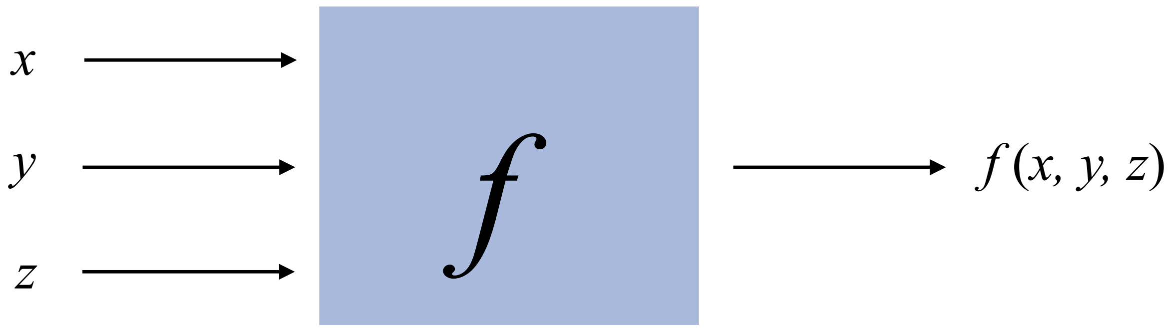
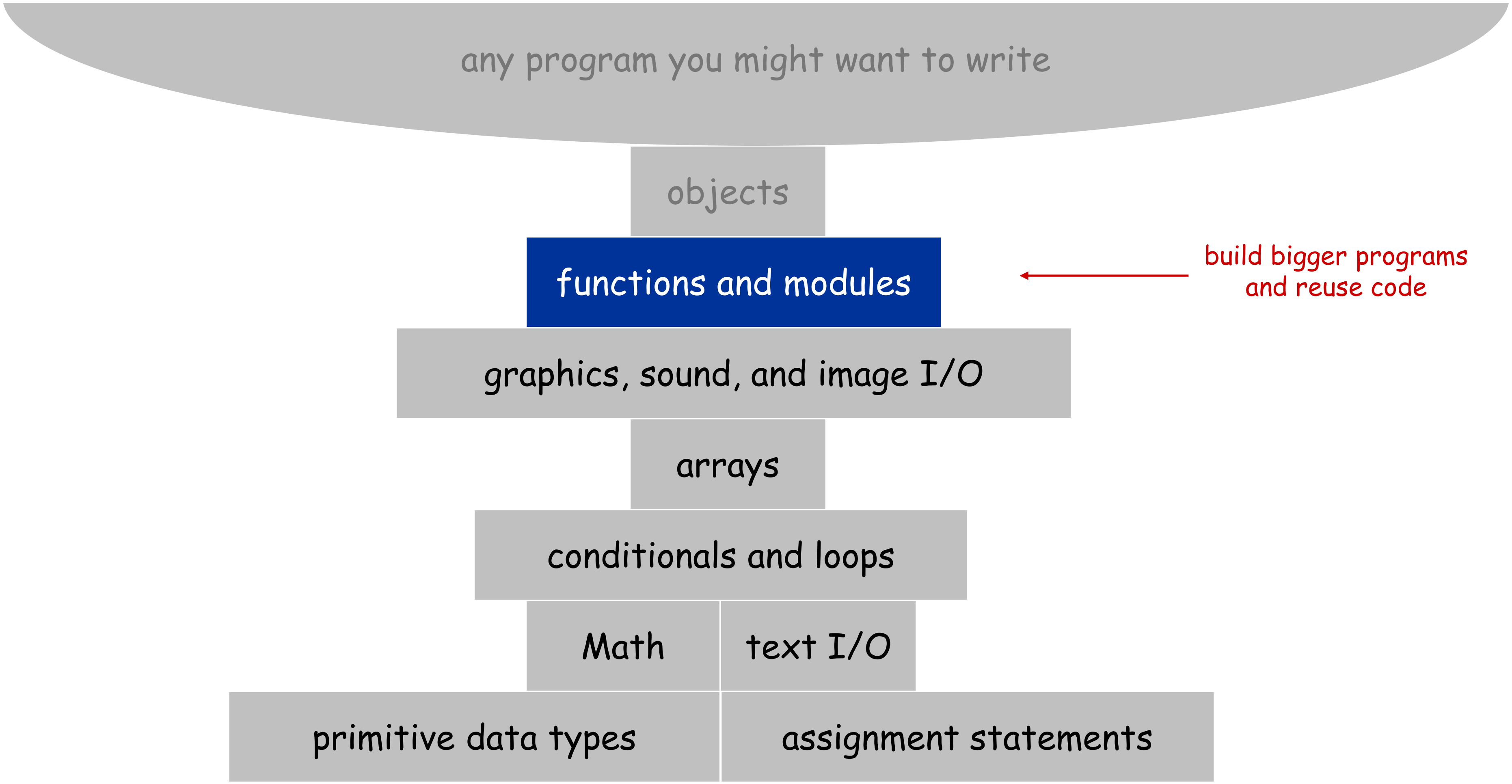
2.1 Functions



2.1 Functions



A Foundation for Programming



Functions (Static Methods)

Java function.

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

Applications.

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

Examples.

- Built-in functions: `Math.random()`, `Math.abs()`, `Integer.parseInt()`.
- 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.

$$2.0 \xrightarrow{\text{input}} \boxed{f(x) = \sqrt{x}} \xrightarrow{\text{output}} 1.414213\ldots$$

signature *return type* *method name* *argument type* *argument variable*

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

local variables

method body

return statement

call on another method

Flow of Control

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

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

   public static void main(String[] args)
   {
      int N = args.length;
      double[] a = new double[N];
      for (int i = 0; i < N; i++)
         a[i] = Double.parseDouble(args[i]);
      for (int i = 0; i < N; i++)
      {
         double x = sqrt(a[i]);
         StdOut.println(x);
      }
   }
}
```

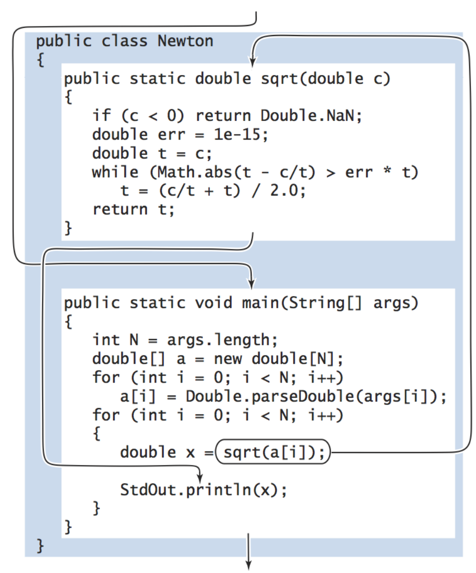
implicit return statement
at end of void function

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.

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


   public static void main(String[] args)
   {
      int N = args.length;
      double[] a = new double[N];
      for (int i = 0; i < N; i++)
         a[i] = Double.parseDouble(args[i]);

      for (int i = 0; i < N; i++)
      {
         double x = sqrt(a[i]);
         StdOut.println(x);
      }
   }
}
```

scope of c, err, *and* t

this code cannot refer to args[], N, *or* a[]

this code cannot refer to c[], err, *or* t

scope of args[], N, *and* a[]

scope of i

two different variables

scope of i *and* x

Best practice: declare variables to limit their scope.

Function Challenge 1a

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

```
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?

```
public class Cubes2 {

   public static int cube(int i) {
      int 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 1c

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

```
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?

```
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));
   }
}
```

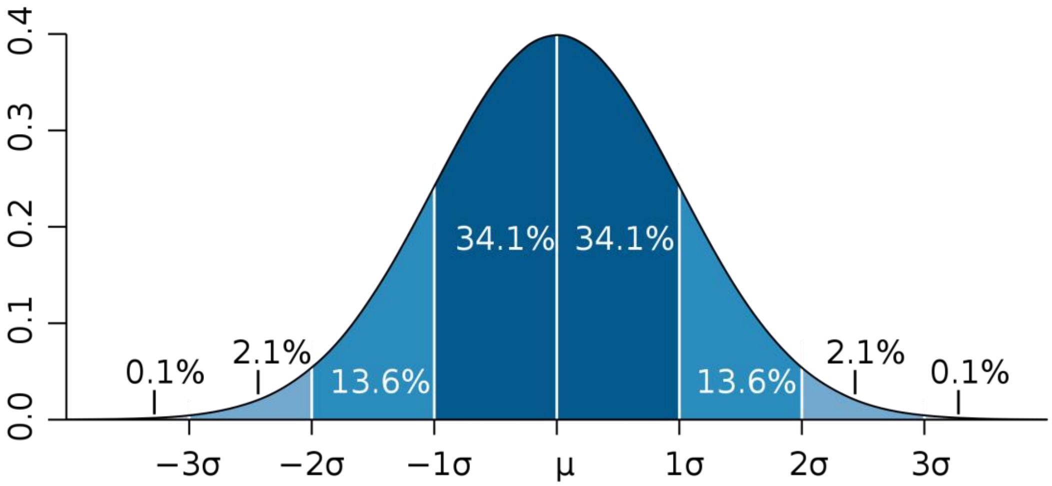
Function Challenge 1e

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

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

Gaussian Distribution

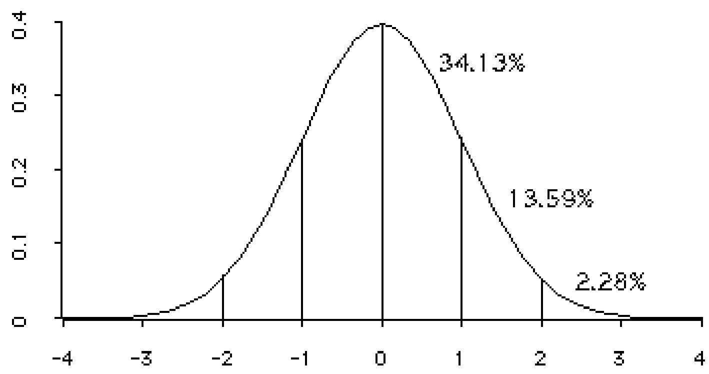


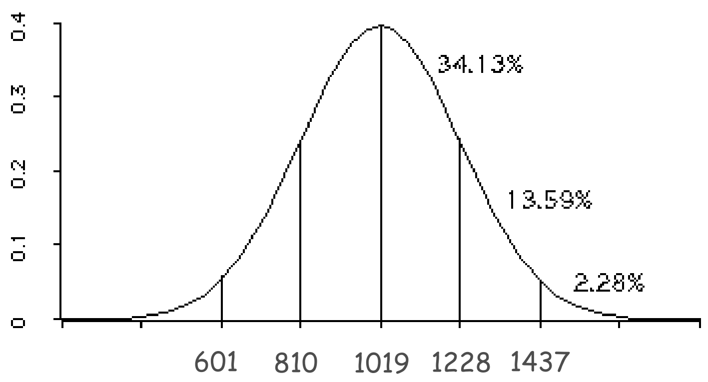
Gaussian Distribution

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 μ = 1019, stddev σ = 209.





$$\phi(x) = \frac{1}{\sqrt{2\pi}}\; e^{-x^2/2}$$

$$\begin{aligned}\phi(x, \mu, \sigma) &= \frac{1}{\sigma\sqrt{2\pi}}\; e^{-(x-\mu)^2 \,/\, 2\sigma^2} \\ &= \phi\left(\frac{x-\mu}{\sigma}\right) / \;\sigma\end{aligned}$$

Java Function for $\phi(x)$

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

$$\phi(x) = \frac{1}{\sqrt{2\pi}}\, e^{-x^2/2}$$

```
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) / sigma;
   }
}
```

$$\phi(x, \mu, \sigma) = \phi\left(\frac{x-\mu}{\sigma}\right) / \sigma$$

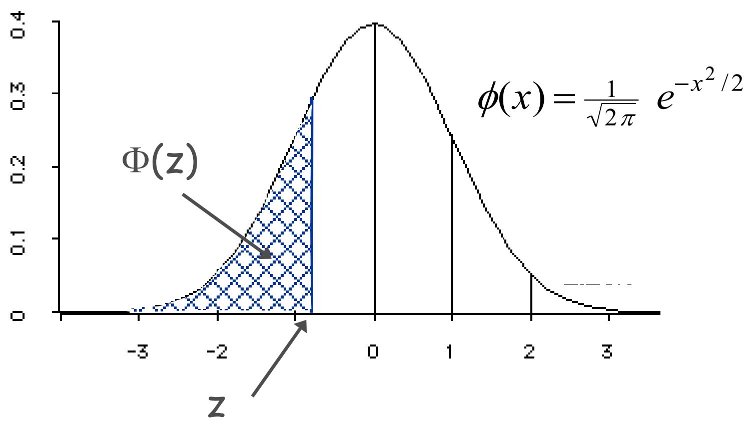
Overloading. Functions with different signatures are different.
Multiple arguments. Functions can take any number of arguments.
Calling other functions. Functions can call other functions.

library or user-defined

Gaussian Cumulative Distribution Function

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

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



Taylor series

$$\begin{aligned} \Phi(z) &= \int_{-\infty}^{z} \phi(x)dx \\ &= \frac{1}{2} + \phi(z) \left(z + \frac{z^3}{3} + \frac{z^5}{3 \cdot 5} + \frac{z^7}{3 \cdot 5 \cdot 7} + \ldots \right) \end{aligned}$$

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

Java function for $\Phi(z)$

```
public class Gaussian {

   public static double phi(double x)
      // as before

   public static double Phi(double z) {
      if (z < -8.0) return 0.0;
      if (z >  8.0) return 1.0;
      double sum = 0.0, term = z;
      for (int i = 3; sum + term != sum; i += 2) {
         sum  = sum + term;
         term = term * z * z / i;
      }
      return 0.5 + sum * phi(z);
   }

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

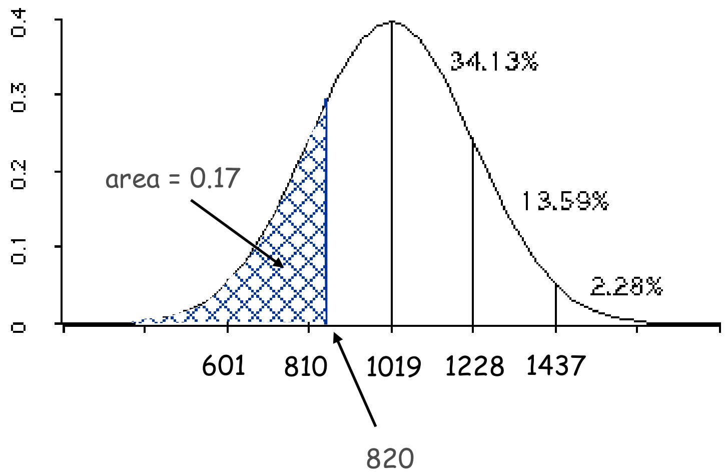
accurate with absolute error less than $8 * 10^{-16}$

$$\Phi(z,\ \mu,\ \sigma)\ =\ \int_{-\infty}^{z} \phi(z, \mu, \sigma)\ =\ \Phi((z-\mu)\ /\ \sigma)$$

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%]




```
double fraction = Gaussian.Phi(820.0, 1019.0, 209.0);
```

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.

- Takes you beyond pre-packaged libraries.
- You build the tools you need: `Gaussian.phi()`, …

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.

Extra Slides

Function Examples

absolute value of an `int` *value*

```
public static int abs(int x)
{
   if (x < 0) return -x;
   else       return  x;
}
```

overloading

absolute value of a `double` *value*

```
public static double abs(double x)
{
   if (x < 0.0) return -x;
   else         return  x;
}
```

primality test

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

multiple arguments

hypotenuse of a right triangle

```
public static double hypotenuse(double a, double b)
{  return Math.sqrt(a*a + b*b);  }
```