

# APIs

`public class System.out/StdOut/Out`

---

|  |                                       |
|--|---------------------------------------|
| <code>Out(String name)</code>            | <i>create output stream from name</i> |
| <code>void print(String s)</code>        | <i>print s</i>                        |
| <code>void println(String s)</code>      | <i>print s, followed by newline</i>   |
| <code>void println()</code>              | <i>print a new line</i>               |
| <code>void printf(String f, ... )</code> | <i>formatted print</i>                |

*Note: Methods are static and constructor does not apply for System.out/StdOut.*

`public class Math`

---

|   |                            |
|---|----------------------------|
| <code>double abs(double a)</code>           | <i>absolute value of a</i> |
| <code>double max(double a, double b)</code> | <i>maximum of a and b</i>  |
| <code>double min(double a, double b)</code> | <i>minimum of a and b</i>  |

*Note 1: abs(), max(), and min() are defined also for int, long, and float.*

|                                       |                         |
|---------------------------------------|-------------------------|
| <code>double sin(double theta)</code> | <i>sine function</i>    |
| <code>double cos(double theta)</code> | <i>cosine function</i>  |
| <code>double tan(double theta)</code> | <i>tangent function</i> |

*Note 2: Angles are expressed in radians. Use toDegrees() and toRadians() to convert.*

*Note 3: Use asin(), acos(), and atan() for inverse functions.*

|   |   |
|---|---|
| <code>double exp(double a)</code>           | <i>exponential (<math>e^a</math>)</i>                             |
| <code>double log(double a)</code>           | <i>natural log (<math>\log_e a</math>, or <math>\ln a</math>)</i> |
| <code>double pow(double a, double b)</code> | <i>raise a to the bth power (<math>a^b</math>)</i>                |
| <code>long round(double a)</code>           | <i>round to the nearest integer</i>                               |
| <code>double random()</code>                | <i>random number in [0, 1)</i>                                    |
| <code>double sqrt(double a)</code>          | <i>square root of a</i>   |
| <code>double E</code>                       | <i>value of e (constant)</i>                                      |
| <code>double PI</code>                      | <i>value of <math>\pi</math> (constant)</i>                       |



`public class StdIn/In`

---

|                                    |   |
|------------------------------------|---|
| <code>In(String name)</code>       | <i>create input stream from name</i>      |
| <code>boolean isEmpty()</code>     | <i>true if no more values, else false</i> |
| <code>int readInt()</code>         | <i>read a value of type int</i>           |
| <code>double readDouble()</code>   | <i>read a value of type double</i>        |
| <code>long readLong()</code>       | <i>read a value of type long</i>          |
| <code>boolean readBoolean()</code> | <i>read a value of type boolean</i>       |
| <code>char readChar()</code>       | <i>read a value of type char</i>          |
| <code>String readString()</code>   | <i>read a value of type String</i>        |
| <code>String readLine()</code>     | <i>read the rest of the line</i>          |
| <code>String readAll()</code>      | <i>read the rest of the text</i>          |

*Note: Methods are static and constructor does not apply for StdIn.*

`public class String`

---

|  |   |
|--|---|
| <code>String(String s)</code>                      | <i>create a string with the same value as s</i> |
| <code>int length()</code>                          | <i>string length</i>                            |
| <code>char charAt(int i)</code>                    | <i>i th character</i>                           |
| <code>String substring(int i, int j)</code>        | <i>i th through (j-1)st characters</i>          |
| <code>boolean contains(String sub)</code>          | <i>does string contain sub as a substring?</i>  |
| <code>boolean startsWith(String pre)</code>        | <i>does string start with pre?</i>              |
| <code>boolean endsWith(String post)</code>         | <i>does string end with post?</i>               |
| <code>int indexOf(String p)</code>                 | <i>index of first occurrence of p</i>           |
| <code>int indexOf(String p, int i)</code>          | <i>index of first occurrence of p after i</i>   |
| <code>String concat(String t)</code>               | <i>this string with t appended</i>              |
| <code>int compareTo(String t)</code>               | <i>string comparison</i>                        |
| <code>String replaceAll(String a, String b)</code> | <i>result of changing as to bs</i>              |
| <code>String[] split(String delim)</code>          | <i>strings between occurrences of delim</i>     |
| <code>boolean equals(String t)</code>              | <i>is this string's value the same as t's?</i>  |

public class StdDraw/Draw

|   |  |
|---|--|
| Draw()  | <i>create a new Draw object</i>                        |
| void line(double x0, double y0, double x1, double y1) |  |
| void point(double x, double y)                        |  |
| void text(double x, double y, String s)               |  |
| void circle(double x, double y, double r)             |  |
| void filledCircle(double x, double y, double r)       |  |
| void square(double x, double y, double r)             |  |
| void filledSquare(double x, double y, double r)       |  |
| void polygon(double[] x, double[] y)                  |  |
| void filledPolygon(double[] x, double[] y)            |  |
| void setXscale(double x0, double x1)                  | <i>reset x range to (x<sub>0</sub>, x<sub>1</sub>)</i> |
| void setYscale(double y0, double y1)                  | <i>reset y range to (y<sub>0</sub>, y<sub>1</sub>)</i> |
| void setPenRadius(double r)                           | <i>set pen radius to r</i>                             |
| void setPenColor(Color c)                             | <i>set pen color to c</i>                              |
| void setFont(Font f)                                  | <i>set text font to f</i>                              |
| void setCanvasSize(int w, int h)                      | <i>set canvas to w-by-h window</i>                     |
| void clear(Color c)                                   | <i>clear the canvas; color it c</i>                    |
| void show(int dt)                                     | <i>show all; pause dt milliseconds</i>                 |
| void save(String filename)                            | <i>save to a .jpg or .png file</i>                     |

*Note: Methods are static and constructor does not apply for StdDraw.*

public class StdAudio

|                                    |                                       |
|------------------------------------|---------------------------------------|
| void play(String file)             | <i>play the given .wav file</i>       |
| void play(double[] a)              | <i>play the given sound wave</i>      |
| void play(double x)                | <i>play sample for 1/44100 second</i> |
| void save(String file, double[] a) | <i>save to a .wav file</i>            |
| double[] read(String file)         | <i>read from a .wav file</i>          |

---

**public class StdRandom**


---

|   |   |
|---|---|
| <code>int uniform(int N)</code>                   | <i>integer between 0 and N-1</i>            |
| <code>double uniform(double lo, double hi)</code> | <i>real between lo and hi</i>               |
| <code>boolean bernoulli(double p)</code>          | <i>true with probability p</i>              |
| <code>double gaussian()</code>                    | <i>normal, mean 0, standard deviation 1</i> |
| <code>double gaussian(double m, double s)</code>  | <i>normal, mean m, standard deviation s</i> |
| <code>int discrete(double[] a)</code>             | <i>i with probability a[i]</i>              |
| <code>void shuffle(double[] a)</code>             | <i>randomly shuffle the array a[]</i>       |

---

**public class StdArrayIO**


---

|  |   |
|--|---|
| <code>double[] readDouble1D()</code>   | <i>read a one-dimensional array of double values</i>  |
| <code>double[][] readDouble2D()</code> | <i>read a two-dimensional array of double values</i>  |
| <code>void print(double[] a)</code>    | <i>print a one-dimensional array of double values</i> |
| <code>void print(double[][] a)</code>  | <i>print a two-dimensional array of double values</i> |

*Note 1. 1D format is an integer N followed by N values.*

*Note 2. 2D format is two integers M and N followed by M×N values in row-major order.*

*Note 3. Methods for int and boolean are also included.*

---

**public class StdStats**


---

|  |   |
|--|---|
| <code>double max(double[] a)</code>      | <i>largest value</i>                    |
| <code>double min(double[] a)</code>      | <i>smallest value</i>                   |
| <code>double mean(double[] a)</code>     | <i>average</i>                          |
| <code>double var(double[] a)</code>      | <i>sample variance</i>                  |
| <code>double stddev(double[] a)</code>   | <i>sample standard deviation</i>        |
| <code>double median(double[] a)</code>   | <i>median</i>                           |
| <code>void plotPoints(double[] a)</code> | <i>plot points at (i, a[i])</i>         |
| <code>void plotLines(double[] a)</code>  | <i>plot lines connecting (i, a[i])</i>  |
| <code>void plotBars(double[] a)</code>   | <i>plot bars to points at (i, a[i])</i> |

*Note: overloaded implementations are included for all numeric types*

---

**public class `Picture`**


---

|  |   |
|--|---|
| <code>Picture(String name)</code>            | <i>create a picture from a file</i>       |
| <code>Picture(int w, int h)</code>           | <i>create a blank w-by-h picture</i>      |
| <code>int width()</code>                     | <i>return the width of the picture</i>    |
| <code>int height()</code>                    | <i>return the height of the picture</i>   |
| <code>Color get(int i, int j)</code>         | <i>return the color of pixel (i, j)</i>   |
| <code>void set(int i, int j, Color c)</code> | <i>set the color of pixel (i, j) to c</i> |
| <code>void show()</code>                     | <i>display the image in a window</i>      |
| <code>void save(String name)</code>          | <i>save the image to a file</i>           |

---

**public class `Stopwatch`**


---

|                                   |   |
|-----------------------------------|---|
| <code>Stopwatch()</code>          | <i>create a new stopwatch and start it running</i>        |
| <code>double elapsedTime()</code> | <i>return the elapsed time since creation, in seconds</i> |

---

**public class `Histogram`**


---

|   |  |
|---|--|
| <code>Histogram(int N)</code>           | <i>create a dynamic histogram for the N integer values in [0, N)</i> |
| <code>double addDataPoint(int i)</code> | <i>add an occurrence of the value i</i>                              |

---

**public class `Turtle`**


---

|  |  |
|--|--|
| <code>Turtle(double x0, double y0, double a0)</code> | <i>create a new turtle at (x<sub>0</sub>, y<sub>0</sub>) facing a<sub>0</sub> degrees counterclockwise from x-axis</i> |
| <code>void turnLeft(double delta)</code>             | <i>rotate delta degrees counterclockwise</i>   |
| <code>void goForward(double step)</code>             | <i>move distance step, drawing a line</i>  |

---

 public class Counter
 

---

|                             |  |
|-----------------------------|--|
| Counter(String id, int max) | <i>create a counter, initialized to 0</i>        |
| void increment()            | <i>increment counter unless its value is max</i> |
| int value()                 | <i>return the value of the counter</i>           |
| String toString()           | <i>string representation</i>                     |

---

 public class Complex
 

---

|                                   |                                     |
|-----------------------------------|-------------------------------------|
| Complex(double real, double imag) |                                     |
| Complex plus(Complex b)           | <i>sum of this number and b</i>     |
| Complex times(Complex b)          | <i>product of this number and b</i> |
| double abs()                      | <i>magnitude</i>                    |
| double re()                       | <i>real part</i>                    |
| double im()                       | <i>imaginary part</i>               |
| String toString()                 | <i>string representation</i>        |

---

 public class Vector
 

---

|                         |   |
|-------------------------|---|
| Vector(double[] a)      | <i>create a vector with the given Cartesian coordinates</i> |
| Vector plus(Vector b)   | <i>sum of this vector and b</i>                             |
| Vector minus(Vector b)  | <i>difference of this vector and b</i>                      |
| Vector times(double t)  | <i>scalar product of this vector and t</i>                  |
| double dot(Vector b)    | <i>dot product of this vector and b</i>                     |
| double magnitude()      | <i>magnitude of this vector</i>                             |
| Vector direction()      | <i>unit vector with same direction as this vector</i>       |
| double cartesian(int i) | <i>i<sup>th</sup> cartesian coordinate of this vector</i>   |
| String toString()       | <i>string representation</i>                                |

```
public class Stack<Item>
```

---

```

    Stack<Item>                create an empty stack
    boolean isEmpty()         is the stack empty?
    void push(Item item)     push an item onto the stack
    Item pop()               pop the stack

```

```
public class Queue<Item>
```

---

```

    Queue<Item>()            create an empty queue
    boolean isEmpty()       is the queue empty?
    void enqueue(Item item) enqueue an item
    Item dequeue()         dequeue an item
    int length()           queue length

```

```
public class ST<Key extends Comparable<Key>, Value>
```

---

```

    ST()                    create a symbol table
    void put(Key key, Value v) put key-value pair into the table
    Value get(Key key)     return value paired with key, null if key not in table
    boolean contains(Key key) is there a value paired with key?

```

```
public class SET<Key extends Comparable<Key>>
```

---

```

    SET()                  create a set
    boolean isEmpty()     is the set empty?
    void add(Key key)     add key to the set
    boolean contains(Key key) is key in the set?

```

public class `Graph`

---

|  |                                     |
|--|-------------------------------------|
| <code>Graph()</code>                                     | <i>create an empty graph</i>        |
| <code>Graph(In in, String delim)</code>                  | <i>read graph from input stream</i> |
| <code>void addEdge(String v, String w)</code>            | <i>add edge v-w</i>                 |
| <code>int V()</code>                                     | <i>number of vertices</i>           |
| <code>int E()</code>                                     | <i>number of edges</i>              |
| <code>Iterable&lt;String&gt; vertices()</code>           | <i>vertices in the graph</i>        |
| <code>Iterable&lt;String&gt; adjacentTo(String v)</code> | <i>neighbors of v</i>               |
| <code>int degree(String v)</code>                        | <i>number of neighbors of v</i>     |
| <code>boolean hasVertex(String v)</code>                 | <i>is v a vertex in the graph?</i>  |
| <code>boolean hasEdge(String v, String w)</code>         | <i>is v-w an edge in the graph?</i> |

public class `PathFinder`

---

|  |  |
|--|--|
| <code>PathFinder(Graph G, String s)</code>           | <i>create an object that finds paths in G from s</i> |
| <code>int distanceTo(String v)</code>                | <i>length of shortest path from s to v in G</i>      |
| <code>Iterable&lt;String&gt; pathTo(String v)</code> | <i>shortest path from s to v in G</i>                |