Software Development (cs2500)

Lecture 11: Iteration (Continued)

M. R. C. van Dongen

October 16, 2013

Software Development

M. R. C. van Dongen

Sentinel Values

break and continue

Common Algorithms

Nested Loops

Simulations

For Friday

Acknowledgements

References

About this Document

Sentinel Values

- Some programs must deal with variable numbers of input.
  - For example, the bank accounts of a bank.
  - You don't know how many accounts there are in advance.
  - You must process them all.
- The program must have some way to stop.
- You could use the number of bank accounts as inputs.
  - This is a bit inconvenient.
- Most such programs stop when they read a special input value.
  - Programs dealing with files use a special *end of file* (EOF) value.
  - Programs dealing with numbers could use "disallowed" numbers.
  - E.g. -1 if the program deals with non-negative numbers.
- Such values are called *sentinels*.

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
For Friday
Acknowledgements
References
About this Document

Computing the Salary

Java

```
public static void main( String[] args ) {
    final Scanner scanner = new Scanner( System.in );

    double sum = 0.0;
    double next = readNextSalary( scanner );
    while (next != -1.0) {
        sum += next;
        next = readNextSalary( scanner );
    }

    System.err.println( "Total salary is " + sum );
}

private static double readNextSalary( final Scanner scanner ) {
    System.err.print( "Enter salaries, -1 to finish: " );
    return scanner.nextDouble( );
}
```

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
For Friday
Acknowledgements
References
About this Document

`break` and `continue`

- `Java` has two more loop-related statements.

`break`	Jumps out of the loop.
`continue`	Jumps to the end of the current loop.

- They are perfectly valid `Java`.
- However, we will not use them for cs2500.
  - Leads to spaghetti code.
  - Makes reasoning about program flow more difficult.
  - Makes if difficult to write proper invariants.
  - Avoiding the statements is a good mental exercise.
  - You will find better, cleaner, more elegant algorithms.
- If you use them, you will lose marks.

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
For Friday
Acknowledgements
References
About this Document

Computing Averages

Java

```
final Scanner scanner = new Scanner( System.in );
int numbersRead = 0;
int total = 0;

while (scanner.hasNextInt( )) {
    total += scanner.nextInt( );
    numbersRead++;
}

final String average;
if (numbersRead != 0) {
    average = Double.toString( total/(double)numbersRead );
} else {
    average = "undefined";
}

System.err.println( "Total: " + total );
System.err.println( "#Iterations: " + numbersRead );
System.err.println( "Average: " + average );
```

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
For Friday
Acknowledgements
References
About this Document

Computing Averages

Java

```
final Scanner scanner = new Scanner( System.in );
int numbersRead = 0;
int total = 0;

while (scanner.hasNextInt( )) {
    total += scanner.nextInt( );
    numbersRead++;
}

final String average;
if (numbersRead != 0) {
    average = Double.toString( total/(double)numbersRead );
} else {
    average = "undefined";
}

System.err.println( "Total: " + total );
System.err.println( "#Iterations: " + numbersRead );
System.err.println( "Average: " + average );
```

Computing the Maximum

We Assume there is Some Input

Java

```
final Scanner scanner = new Scanner( System.in );
int currentMax = Integer.MIN_VALUE;

do {
    final int next = scanner.nextInt( );
    currentMax = currentMax < next ? next : currentMax;
} while (scanner.hasNextInt( ));

System.err.println( "Maximum: " + currentMax );
```

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
For Friday
Acknowledgements
References
About this Document

Computing the Maximum

We Assume there is Some Input

Java

```
final Scanner scanner = new Scanner( System.in );
int currentMax = Integer.MIN_VALUE;

do {
    final int next = scanner.nextInt( );
    currentMax = currentMax < next ? next : currentMax;
} while (scanner.hasNextInt( ));

System.err.println( "Maximum: " + currentMax );
```

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
For Friday
Acknowledgements
References
About this Document

Computing the Maximum

We Assume there is Some Input

Java

```
final Scanner scanner = new Scanner( System.in );
int currentMax = Integer.MIN_VALUE;

do {
    final int next = scanner.nextInt( );
    currentMax = Integer.max( next, currentMax );
} while (scanner.hasNextInt( ));

System.err.println( "Maximum: " + currentMax );
```

Sentinel Values

break and continue

Common Algorithms

Nested Loops

Simulations

For Friday

Acknowledgements

References

About this Document

Number Conversion

Java

```java
private static final int BASE = 10;

public static String toString( final int value ) {
    String absStringValue = "";
    final boolean negative = value < 0;
    int residue = value;

    while ((residue <= -BASE) || (BASE <= residue)) {
        // strip off last digit.
        final int lastDigit = residue % BASE;
        // add last digit.
        absStringValue = makeDecimalString( lastDigit ) + absStringValue;
        // get rid of last digit.
        residue = residue / BASE;
    }
    // add remaining digit.
    absStringValue = makeDecimalString( residue ) + absStringValue;

    // add sign if needed.
    return (negative ? "-" : "") + absStringValue;
}
```

Sentinel Values

break and continue

Common Algorithms

Nested Loops

Simulations

For Friday

Acknowledgements

References

About this Document

Number Conversion

Java

```
private static final int BASE = 10;

public static String toString( final int value ) {
    String absStringValue = "";
    final boolean negative = value < 0;
    int residue = value;

    while ((residue <= -BASE) || (BASE <= residue)) {
        // strip off last digit.
        final int lastDigit = residue % BASE;
        // add last digit.
        absStringValue = makeDecimalString( lastDigit ) + absStringValue;
        // get rid of last digit.
        residue = residue / BASE;
    }
    // add remaining digit.
    absStringValue = makeDecimalString( residue ) + absStringValue;

    // add sign if needed.
    return (negative ? "-" : "") + absStringValue;
}
```

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
For Friday
Acknowledgements
References
About this Document

Number Conversion

Java

```java
private static final int BASE = 10;

public static String toString( final int value ) {
    String absStringValue = "";
    final boolean negative = value < 0;
    int residue = value;

    while ((residue <= -BASE) || (BASE <= residue)) {
        // strip off last digit.
        final int lastDigit = residue % BASE;
        // add last digit.
        absStringValue = makeDecimalString( lastDigit ) + absStringValue;
        // get rid of last digit.
        residue = residue / BASE;
    }
    // add remaining digit.
    absStringValue = makeDecimalString( residue ) + absStringValue;

    // add sign if needed.
    return (negative ? "-" : "") + absStringValue;
}
```

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
For Friday
Acknowledgements
References
About this Document

Number Conversion

Java

```
private static final int BASE = 10;

public static String toString( final int value ) {
    String absStringValue = "";
    final boolean negative = value < 0;
    int residue = value;

    while ((residue <= -BASE) || (BASE <= residue)) {
        // strip off last digit.
        final int lastDigit = residue % BASE;
        // add last digit.
        absStringValue = makeDecimalString( lastDigit ) + absStringValue;
        // get rid of last digit.
        residue = residue / BASE;
    }
    // add remaining digit.
    absStringValue = makeDecimalString( residue ) + absStringValue;

    // add sign if needed.
    return (negative ? "-" : "") + absStringValue;
}
```

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
For Friday
Acknowledgements
References
About this Document

Number Conversion

Java

```
private static final int BASE = 10;

public static String toString( final int value ) {
    String absStringValue = "";
    final boolean negative = value < 0;
    int residue = value;

    while ((residue <= -BASE) || (BASE <= residue)) {
        // strip off last digit.
        final int lastDigit = residue % BASE;
        // add last digit.
        absStringValue = makeDecimalString( lastDigit ) + absStringValue;
        // get rid of last digit.
        residue = residue / BASE;
    }
    // add remaining digit.
    absStringValue = makeDecimalString( residue ) + absStringValue;

    // add sign if needed.
    return (negative ? "-" : "") + absStringValue;
}
```

Number Conversion

Java

```
private static final int BASE = 10;

public static String toString( final int value ) {
    String absStringValue = "";
    final boolean negative = value < 0;
    int residue = value;

    while ((residue <= -BASE) || (BASE <= residue)) {
        // strip off last digit.
        final int lastDigit = residue % BASE;
        // add last digit.
        absStringValue = makeDecimalString( lastDigit ) + absStringValue;
        // get rid of last digit.
        residue = residue / BASE;
    }
    // add remaining digit.
    absStringValue = makeDecimalString( residue ) + absStringValue;

    // add sign if needed.
    return (negative ? "-" : "") + absStringValue;
}
```

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
For Friday
Acknowledgements
References
About this Document

Number Conversion

Java

```
private static final int BASE = 10;

public static String toString( final int value ) {
    String absStringValue = "";
    final boolean negative = value < 0;
    int residue = value;

    while ((residue <= -BASE) || (BASE <= residue)) {
        // strip off last digit.
        final int lastDigit = residue % BASE;
        // add last digit.
        absStringValue = makeDecimalString( lastDigit ) + absStringValue;
        // get rid of last digit.
        residue = residue / BASE;
    }
    // add remaining digit.
    absStringValue = makeDecimalString( residue ) + absStringValue;

    // add sign if needed.
    return (negative ? "-" : "") + absStringValue;
}
```

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
For Friday
Acknowledgements
References
About this Document

Number Conversion

Java

```java
private static final int BASE = 10;

public static String toString( final int value ) {
    String absStringValue = "";
    final boolean negative = value < 0;
    int residue = value;

    while ((residue <= -BASE) || (BASE <= residue)) {
        // strip off last digit.
        final int lastDigit = residue % BASE;
        // add last digit.
        absStringValue = makeDecimalString( lastDigit ) + absStringValue;
        // get rid of last digit.
        residue = residue / BASE;
    }
    // add remaining digit.
    absStringValue = makeDecimalString( residue ) + absStringValue;

    // add sign if needed.
    return (negative ? "-" : "") + absStringValue;
}
```

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
For Friday
Acknowledgements
References
About this Document

Number Conversion

Java

```
private static String makeDecimalString( final int digit ) {
    final char character = (char)(Math.abs( digit ) + (int)'0');
    final char[] characterArray = { character };
    return new String( characterArray );
}
```

Sentinel Values

break and continue

Common Algorithms

Nested Loops

Simulations

For Friday

Acknowledgements

References

About this Document

Number Conversion

Java

```
private static String makeDecimalString( final int digit ) {
    final char character = (char)(Math.abs( digit ) + (int)'0');
    final char[] characterArray = { character };
    return new String( characterArray );
}
```

Sentinel Values

break and continue

Common Algorithms

Nested Loops

Simulations

For Friday

Acknowledgements

References

About this Document

Number Conversion

Java

```java
private static String makeDecimalString( final int digit ) {
    final char character = (char)(Math.abs( digit ) + (int)'0');
    final char[] characterArray = { character };
    return new String( characterArray );
}
```

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
For Friday
Acknowledgements
References
About this Document

Number Conversion

Java

```
private static String makeDecimalString( final int digit ) {
    final char character = (char)(Math.abs( digit ) + (int)'0');
    final char[] characterArray = { character };
    return new String( characterArray );
}
```

Sentinel Values

break and continue

Common Algorithms

Nested Loops

Simulations

For Friday

Acknowledgements

References

About this Document

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
For Friday
Acknowledgements
References
About this Document

Number Conversion

Java

```
private static String makeDecimalString( final int digit ) {
    final char character = (char)(Math.abs( digit ) + (int)'0');
    final char[] characterArray = { character };
    return new String( characterArray );
}
```

Nested Loops

- Printing columns of the rows in a spreadsheet.
- Colouring pixels at x- and y-coordinates of a picture.
- ...

Sentinel Values
`break` and `continue`
Common Algorithms
Nested Loops
Simulations
For Friday
Acknowledgements
References
About this Document

Power Table

Unix Session

```
$
```

Sentinel Values

break and continue

Common Algorithms

Nested Loops

Simulations

For Friday

Acknowledgements

References

About this Document

Power Table

Unix Session

```
$ java PowerTable
```

Sentinel Values

break and continue

Common Algorithms

Nested Loops

Simulations

For Friday

Acknowledgements

References

About this Document

Power Table

Unix Session

```
$ java PowerTable
Enter maximum power:
```

Power Table

Unix Session

```
$ java PowerTable
Enter maximum power: 4
```

Software Development

M. R. C. van Dongen

Sentinel Values

break and continue

Common Algorithms

Nested Loops

Simulations

For Friday

Acknowledgements

References

About this Document

Power Table

Unix Session

```
$ java PowerTable
Enter maximum power: 4
Enter last integer:
```

Sentinel Values

break and continue

Common Algorithms

Nested Loops

Simulations

For Friday

Acknowledgements

References

About this Document

Power Table

Unix Session

```
$ java PowerTable
Enter maximum power: 4
Enter last integer: 6
```

Power Table

Unix Session

```
$ java PowerTable
Enter maximum power: 4
Enter last integer: 6
     1     2     3     4
    x     x     x     x
------------------------
    1     1     1     1
    2     4     8    16
    3     9    27    81
    4    16    64   256
    5    25   125   625
    6    36   216  1296
```

Software Development

M. R. C. van Dongen

Sentinel Values

break and continue

Common Algorithms

Nested Loops

Simulations

For Friday

Acknowledgements

References

About this Document

The main

Java

```
public static void main( String[] args ) {
    final Scanner scanner = new Scanner( System.in );
    System.err.print( "Enter maximum power: " );
    final int maxPower = scanner.nextInt( );
    System.err.print( "Enter last integer: " );
    final int lastInteger = scanner.nextInt( );

    printColumnHeadings( maxPower );
    printRows( lastInteger, maxPower );
}
```

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
For Friday
Acknowledgements
References
About this Document

The main

Java

```
public static void main( String[] args ) {
    final Scanner scanner = new Scanner( System.in );
    System.err.print( "Enter maximum power: " );
    final int maxPower = scanner.nextInt( );
    System.err.print( "Enter last integer: " );
    final int lastInteger = scanner.nextInt( );

    printColumnHeadings( maxPower );
    printRows( lastInteger, maxPower );
}
```

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
For Friday
Acknowledgements
References
About this Document

printColumnHeadings( )

Java

```
private static void printColumnHeadings( final int maxPower ) {
    for (int power = 1; power <= maxPower; power++) {
        System.err.print( String.format( "  %4d", power ) );
    }
    System.err.println( );
    for (int power = 1; power <= maxPower; power++) {
        System.err.print( String.format( " %4s ", "x" ) );
    }
    System.err.println( );
    for (int power = 1; power <= maxPower; power++) {
        System.err.print( "------" );
    }
    System.err.println( );
}
```

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
For Friday
Acknowledgements
References
About this Document

printRows( )

Java

```java
private static final void printRows( final int maxRow,
                                     final int maxPower ) {
    for (int row = 1; row <= maxRow; row++) {
        for (int power = 1; power <= maxPower; power++) {
            final int rowPower = Math.pow( row, power );
            System.err.print( String.format( " %4d ", rowPower ) );
        }
        System.err.println( );
    }
}
```

printRows( )

Java

```
private static final void printRows( final int maxRow,
                                     final int maxPower ) {
    for (int row = 1; row <= maxRow; row++) {
        for (int power = 1; power <= maxPower; power++) {
            final int rowPower = Math.pow( row, power );
            System.err.print( String.format( " %4d ", rowPower ) );
        }
        System.err.println( );
    }
}
```

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
For Friday
Acknowledgements
References
About this Document

printRows( )

Java

```
private static final void printRows( final int maxRow,
                                     final int maxPower ) {
    for (int row = 1; row <= maxRow; row++) {
        for (int power = 1; power <= maxPower; power++) {
            final int rowPower = computePower( row, power );
            System.err.print( String.format( " %4d ", rowPower ) );
        }
        System.err.println( );
    }
}
```

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
For Friday
Acknowledgements
References
About this Document

computePower( )

Java

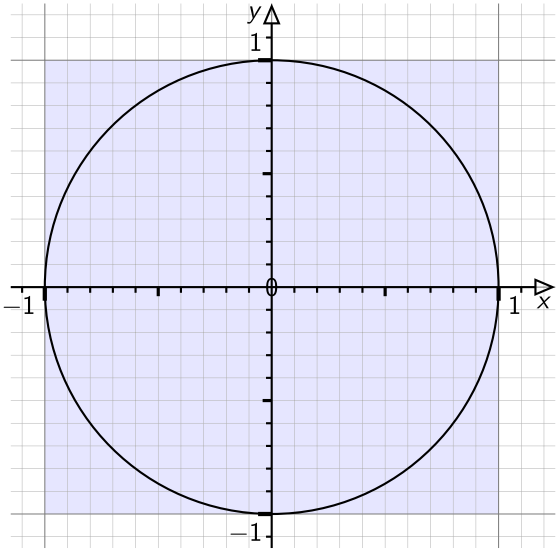
```
private static int computePower( final int number, final int power ) {
    int result = 1;

    for (int product = 1; product <= power; product++) {
        result = result * number;
    }

    return result;
}
```

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
For Friday
Acknowledgements
References
About this Document

Monte Carlo Integration



Sentinel Values

break and continue

Common Algorithms

Nested Loops

Simulations

The Area of a Circle

Implementation

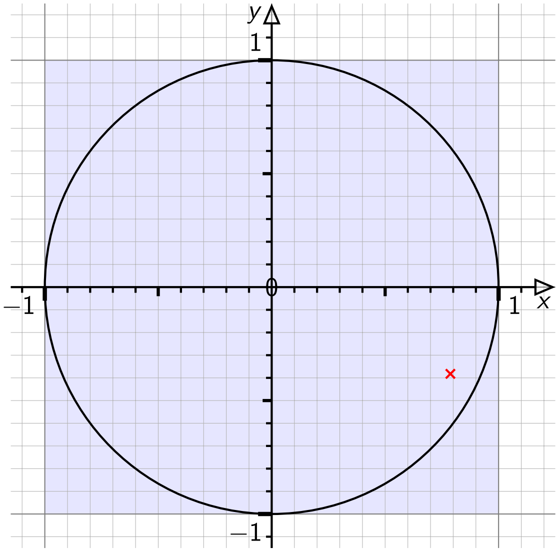
For Friday

Acknowledgements

References

About this Document

Throwing Random Darts



Sentinel Values

break and continue

Common Algorithms

Nested Loops

Simulations

The Area of a Circle

Implementation

For Friday

Acknowledgements

References

About this Document

Sentinel Values

break and continue

Common Algorithms

Nested Loops

Simulations

The Area of a Circle

Implementation

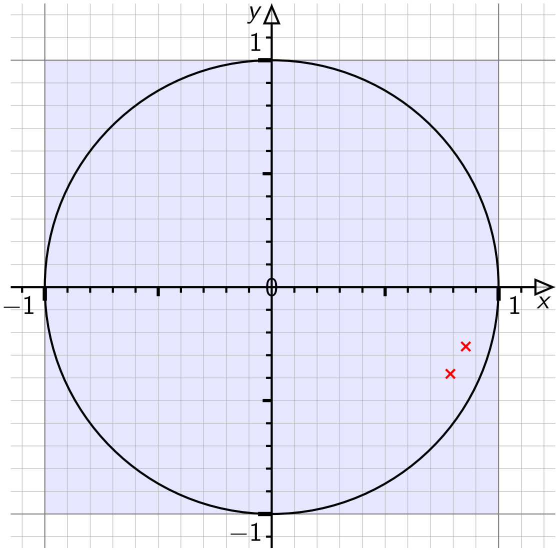
For Friday

Acknowledgements

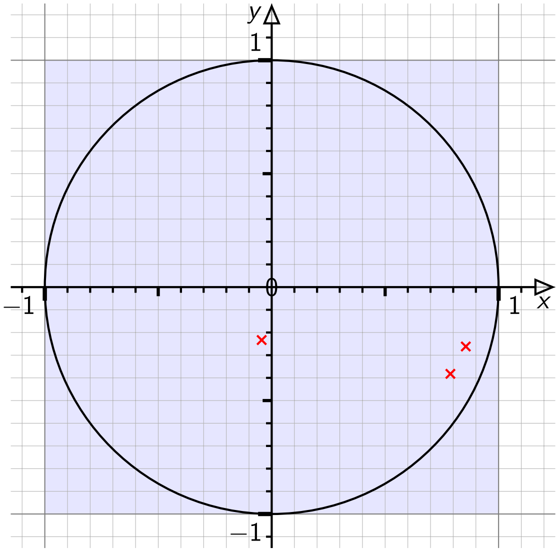
References

About this Document

Throwing Random Darts



Throwing Random Darts



Software Development

M. R. C. van Dongen

Sentinel Values

break and continue

Common Algorithms

Nested Loops

Simulations

The Area of a Circle

Implementation

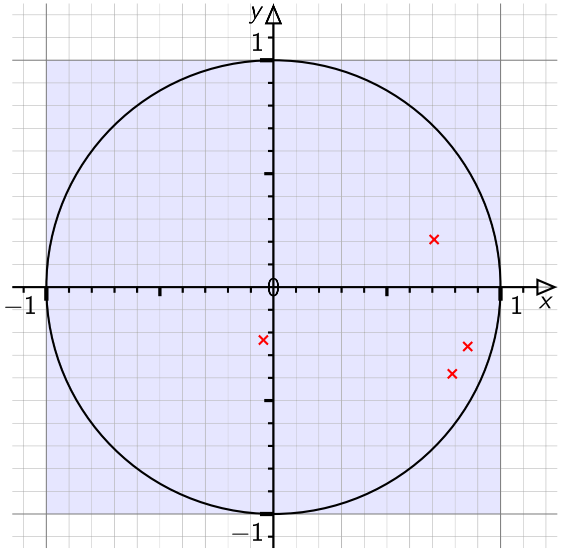
For Friday

Acknowledgements

References

About this Document

Throwing Random Darts



Software Development

M. R. C. van Dongen

Sentinel Values

break and continue

Common Algorithms

Nested Loops

Simulations

The Area of a Circle

Implementation

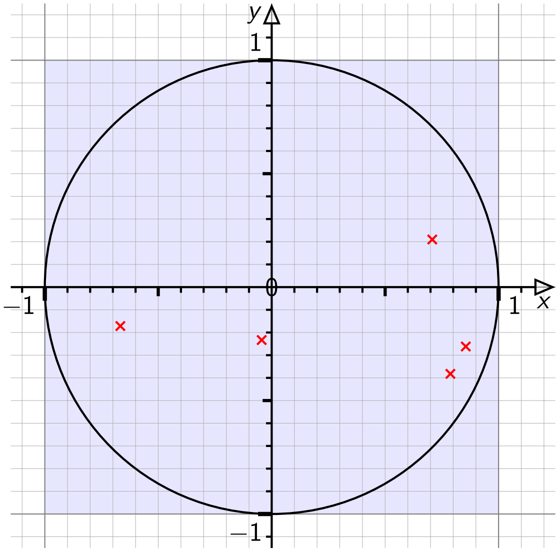
For Friday

Acknowledgements

References

About this Document

Throwing Random Darts



Sentinel Values

break and continue

Common Algorithms

Nested Loops

Simulations

The Area of a Circle

Implementation

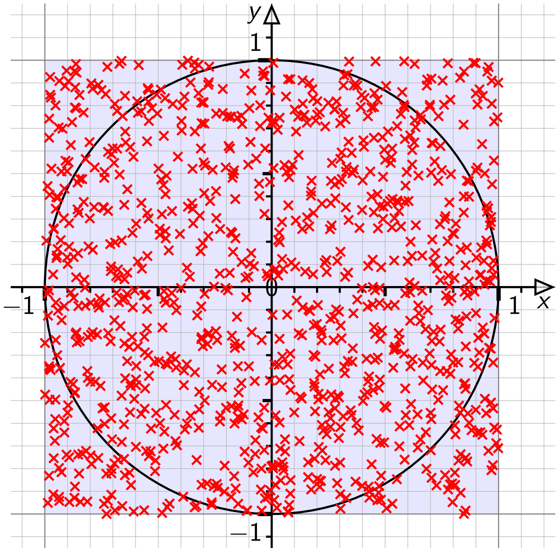
For Friday

Acknowledgements

References

About this Document

Throwing Random Darts



Sentinel Values

`break` and `continue`

Common Algorithms

Nested Loops

Simulations

The Area of a Circle

Implementation

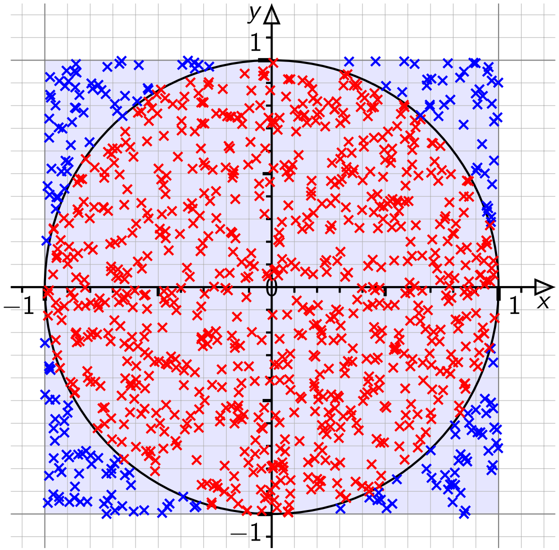
For Friday

Acknowledgements

References

About this Document

Throwing Random Darts



Software Development

M. R. C. van Dongen

Sentinel Values

break and continue

Common Algorithms

Nested Loops

Simulations

The Area of a Circle

Implementation

For Friday

Acknowledgements

References

About this Document

Implementation

Java

```
import java.util.Random;
public class MonteCarlo {
    // Total number of random points.
    private static final int TOTAL_SAMPLES = 1000000;
    // Radius of base circle.
    private static final double RADIUS = 1.0;
    // Total area of extended square.
    private static final double TOTAL_AREA = 4.0 * RADIUS;

    public static void main( String[] args ) {
        final Random rand = new Random( );
        // Initialise number of random points inside circle.
        int hits = 0;
        for ( int sample = TOTAL_SAMPLES; sample-- != 0; ) {
            // Generate next random point.
            final double x = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            final double y = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            // Increment hits if point is inside circle.
            hits += x*x + y*y <= RADIUS * RADIUS ? 1 : 0;
        }
        // Compute approximation of area of circle.
        final double ratio = (double)hits / TOTAL_SAMPLES;
        final double approximation = TOTAL_AREA * ratio;
        System.out.println( "PI = " + Math.PI );
        System.out.println( "PI ~ " + approximation );
    }
}
```

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
The Area of a Circle
Implementation
For Friday
Acknowledgements
References
About this Document

Implementation

Java

```java
import java.util.Random;
public class MonteCarlo {
    // Total number of random points.
    private static final int TOTAL_SAMPLES = 1000000;
    // Radius of base circle.
    private static final double RADIUS = 1.0;
    // Total area of extended square.
    private static final double TOTAL_AREA = 4.0 * RADIUS;

    public static void main( String[] args ) {
        final Random rand = new Random( );
        // Initialise number of random points inside circle.
        int hits = 0;
        for ( int sample = TOTAL_SAMPLES; sample-- != 0; ) {
            // Generate next random point.
            final double x = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            final double y = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            // Increment hits if point is inside circle.
            hits += x*x + y*y <= RADIUS * RADIUS ? 1 : 0;
        }
        // Compute approximation of area of circle.
        final double ratio = (double)hits / TOTAL_SAMPLES;
        final double approximation = TOTAL_AREA * ratio;
        System.out.println( "PI = " + Math.PI );
        System.out.println( "PI ~ " + approximation );
    }
}
```

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
The Area of a Circle
Implementation
For Friday
Acknowledgements
References
About this Document

Implementation

Java

```
import java.util.Random;
public class MonteCarlo {
    // Total number of random points.
    private static final int TOTAL_SAMPLES = 1000000;
    // Radius of base circle.
    private static final double RADIUS = 1.0;
    // Total area of extended square.
    private static final double TOTAL_AREA = 4.0 * RADIUS;

    public static void main( String[] args ) {
        final Random rand = new Random( );
        // Initialise number of random points inside circle.
        int hits = 0;
        for ( int sample = TOTAL_SAMPLES; sample-- != 0; ) {
            // Generate next random point.
            final double x = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            final double y = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            // Increment hits if point is inside circle.
            hits += x*x + y*y <= RADIUS * RADIUS ? 1 : 0;
        }
        // Compute approximation of area of circle.
        final double ratio = (double)hits / TOTAL_SAMPLES;
        final double approximation = TOTAL_AREA * ratio;
        System.out.println( "PI = " + Math.PI );
        System.out.println( "PI ˜ " + approximation );
    }
}
```

Software Development

M. R. C. van Dongen

Sentinel Values

break and continue

Common Algorithms

Nested Loops

Simulations

The Area of a Circle

Implementation

For Friday

Acknowledgements

References

About this Document

Implementation

Java

```java
import java.util.Random;
public class MonteCarlo {
    // Total number of random points.
    private static final int TOTAL_SAMPLES = 1000000;
    // Radius of base circle.
    private static final double RADIUS = 1.0;
    // Total area of extended square.
    private static final double TOTAL_AREA = 4.0 * RADIUS;

    public static void main( String[] args ) {
        final Random rand = new Random( );
        // Initialise number of random points inside circle.
        int hits = 0;
        for ( int sample = TOTAL_SAMPLES; sample-- != 0; ) {
            // Generate next random point.
            final double x = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            final double y = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            // Increment hits if point is inside circle.
            hits += x*x + y*y <= RADIUS * RADIUS ? 1 : 0;
        }
        // Compute approximation of area of circle.
        final double ratio = (double)hits / TOTAL_SAMPLES;
        final double approximation = TOTAL_AREA * ratio;
        System.out.println( "PI = " + Math.PI );
        System.out.println( "PI ~ " + approximation );
    }
}
```

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
The Area of a Circle
Implementation
For Friday
Acknowledgements
References
About this Document

Implementation

Java

```
import java.util.Random;
public class MonteCarlo {
    // Total number of random points.
    private static final int TOTAL_SAMPLES = 1000000;
    // Radius of base circle.
    private static final double RADIUS = 1.0;
    // Total area of extended square.
    private static final double TOTAL_AREA = 4.0 * RADIUS;

    public static void main( String[] args ) {
        final Random rand = new Random( );
        // Initialise number of random points inside circle.
        int hits = 0;
        for ( int sample = TOTAL_SAMPLES; sample-- != 0; ) {
            // Generate next random point.
            final double x = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            final double y = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            // Increment hits if point is inside circle.
            hits += x*x + y*y <= RADIUS * RADIUS ? 1 : 0;
        }
        // Compute approximation of area of circle.
        final double ratio = (double)hits / TOTAL_SAMPLES;
        final double approximation = TOTAL_AREA * ratio;
        System.out.println( "PI = " + Math.PI );
        System.out.println( "PI ~ " + approximation );
    }
}
```

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
The Area of a Circle
Implementation
For Friday
Acknowledgements
References
About this Document

Implementation

Java

```
import java.util.Random;
public class MonteCarlo {
    // Total number of random points.
    private static final int TOTAL_SAMPLES = 1000000;
    // Radius of base circle.
    private static final double RADIUS = 1.0;
    // Total area of extended square.
    private static final double TOTAL_AREA = 4.0 * RADIUS;

    public static void main( String[] args ) {
        final Random rand = new Random( );
        // Initialise number of random points inside circle.
        int hits = 0;
        for ( int sample = TOTAL_SAMPLES; sample-- != 0; ) {
            // Generate next random point.
            final double x = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            final double y = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            // Increment hits if point is inside circle.
            hits += x*x + y*y <= RADIUS * RADIUS ? 1 : 0;
        }
        // Compute approximation of area of circle.
        final double ratio = (double)hits / TOTAL_SAMPLES;
        final double approximation = TOTAL_AREA * ratio;
        System.out.println( "PI = " + Math.PI );
        System.out.println( "PI ˜ " + approximation );
    }
}
```

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
The Area of a Circle
Implementation
For Friday
Acknowledgements
References
About this Document

Implementation

Java

```java
import java.util.Random;
public class MonteCarlo {
    // Total number of random points.
    private static final int TOTAL_SAMPLES = 1000000;
    // Radius of base circle.
    private static final double RADIUS = 1.0;
    // Total area of extended square.
    private static final double TOTAL_AREA = 4.0 * RADIUS;

    public static void main( String[] args ) {
        final Random rand = new Random( );
        // Initialise number of random points inside circle.
        int hits = 0;
        for ( int sample = TOTAL_SAMPLES; sample-- != 0; ) {
            // Generate next random point.
            final double x = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            final double y = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            // Increment hits if point is inside circle.
            hits += x*x + y*y <= RADIUS * RADIUS ? 1 : 0;
        }
        // Compute approximation of area of circle.
        final double ratio = (double)hits / TOTAL_SAMPLES;
        final double approximation = TOTAL_AREA * ratio;
        System.out.println( "PI = " + Math.PI );
        System.out.println( "PI ˜ " + approximation );
    }
}
```

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
The Area of a Circle
Implementation
For Friday
Acknowledgements
References
About this Document

Implementation

Java

```java
import java.util.Random;
public class MonteCarlo {
    // Total number of random points.
    private static final int TOTAL_SAMPLES = 1000000;
    // Radius of base circle.
    private static final double RADIUS = 1.0;
    // Total area of extended square.
    private static final double TOTAL_AREA = 4.0 * RADIUS;

    public static void main( String[] args ) {
        final Random rand = new Random( );
        // Initialise number of random points inside circle.
        int hits = 0;
        for ( int sample = TOTAL_SAMPLES; sample-- != 0; ) {
            // Generate next random point.
            final double x = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            final double y = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            // Increment hits if point is inside circle.
            hits += x*x + y*y <= RADIUS * RADIUS ? 1 : 0;
        }
        // Compute approximation of area of circle.
        final double ratio = (double)hits / TOTAL_SAMPLES;
        final double approximation = TOTAL_AREA * ratio;
        System.out.println( "PI = " + Math.PI );
        System.out.println( "PI ~ " + approximation );
    }
}
```

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
The Area of a Circle
Implementation
For Friday
Acknowledgements
References
About this Document

Implementation

Java

```
import java.util.Random;
public class MonteCarlo {
    // Total number of random points.
    private static final int TOTAL_SAMPLES = 1000000;
    // Radius of base circle.
    private static final double RADIUS = 1.0;
    // Total area of extended square.
    private static final double TOTAL_AREA = 4.0 * RADIUS;

    public static void main( String[] args ) {
        final Random rand = new Random( );
        // Initialise number of random points inside circle.
        int hits = 0;
        for ( int sample = TOTAL_SAMPLES; sample-- != 0; ) {
            // Generate next random point.
            final double x = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            final double y = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            // Increment hits if point is inside circle.
            hits += x*x + y*y <= RADIUS * RADIUS ? 1 : 0;
        }
        // Compute approximation of area of circle.
        final double ratio = (double)hits / TOTAL_SAMPLES;
        final double approximation = TOTAL_AREA * ratio;
        System.out.println( "PI = " + Math.PI );
        System.out.println( "PI ~ " + approximation );
    }
}
```

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
The Area of a Circle
Implementation
For Friday
Acknowledgements
References
About this Document

Implementation

Java

```java
import java.util.Random;
public class MonteCarlo {
    // Total number of random points.
    private static final int TOTAL_SAMPLES = 1000000;
    // Radius of base circle.
    private static final double RADIUS = 1.0;
    // Total area of extended square.
    private static final double TOTAL_AREA = 4.0 * RADIUS;

    public static void main( String[] args ) {
        final Random rand = new Random( );
        // Initialise number of random points inside circle.
        int hits = 0;
        for ( int sample = TOTAL_SAMPLES; sample-- != 0; ) {
            // Generate next random point.
            final double x = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            final double y = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            // Increment hits if point is inside circle.
            hits += x*x + y*y <= RADIUS * RADIUS ? 1 : 0;
        }
        // Compute approximation of area of circle.
        final double ratio = (double)hits / TOTAL_SAMPLES;
        final double approximation = TOTAL_AREA * ratio;
        System.out.println( "PI = " + Math.PI );
        System.out.println( "PI ~ " + approximation );
    }
}
```

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
The Area of a Circle
Implementation
For Friday
Acknowledgements
References
About this Document

Implementation

Java

```
import java.util.Random;
public class MonteCarlo {
    // Total number of random points.
    private static final int TOTAL_SAMPLES = 1000000;
    // Radius of base circle.
    private static final double RADIUS = 1.0;
    // Total area of extended square.
    private static final double TOTAL_AREA = 4.0 * RADIUS;

    public static void main( String[] args ) {
        final Random rand = new Random( );
        // Initialise number of random points inside circle.
        int hits = 0;
        for ( int sample = TOTAL_SAMPLES; sample-- != 0; ) {
            // Generate next random point.
            final double x = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            final double y = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            // Increment hits if point is inside circle.
            hits += x*x + y*y <= RADIUS * RADIUS ? 1 : 0;
        }
        // Compute approximation of area of circle.
        final double ratio = (double)hits / TOTAL_SAMPLES;
        final double approximation = TOTAL_AREA * ratio;
        System.out.println( "PI = " + Math.PI );
        System.out.println( "PI ~ " + approximation );
    }
}
```

Sentinel Values

break and continue

Common Algorithms

Nested Loops

Simulations

The Area of a Circle

Implementation

For Friday

Acknowledgements

References

About this Document

Implementation

Java

```
import java.util.Random;
public class MonteCarlo {
    // Total number of random points.
    private static final int TOTAL_SAMPLES = 1000000;
    // Radius of base circle.
    private static final double RADIUS = 1.0;
    // Total area of extended square.
    private static final double TOTAL_AREA = 4.0 * RADIUS;

    public static void main( String[] args ) {
        final Random rand = new Random( );
        // Initialise number of random points inside circle.
        int hits = 0;
        for ( int sample = TOTAL_SAMPLES; sample-- != 0; ) {
            // Generate next random point.
            final double x = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            final double y = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            // Increment hits if point is inside circle.
            hits += x*x + y*y <= RADIUS * RADIUS ? 1 : 0;
        }
        // Compute approximation of area of circle.
        final double ratio = (double)hits / TOTAL_SAMPLES;
        final double approximation = TOTAL_AREA * ratio;
        System.out.println( "PI = " + Math.PI );
        System.out.println( "PI ˜ " + approximation );
    }
}
```

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
The Area of a Circle
Implementation
For Friday
Acknowledgements
References
About this Document

Implementation

Java

```
import java.util.Random;
public class MonteCarlo {
    // Total number of random points.
    private static final int TOTAL_SAMPLES = 1000000;
    // Radius of base circle.
    private static final double RADIUS = 1.0;
    // Total area of extended square.
    private static final double TOTAL_AREA = 4.0 * RADIUS;

    public static void main( String[] args ) {
        final Random rand = new Random( );
        // Initialise number of random points inside circle.
        int hits = 0;
        for ( int sample = TOTAL_SAMPLES; sample-- != 0; ) {
            // Generate next random point.
            final double x = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            final double y = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            // Increment hits if point is inside circle.
            hits += x*x + y*y <= RADIUS * RADIUS ? 1 : 0;
        }
        // Compute approximation of area of circle.
        final double ratio = (double)hits / TOTAL_SAMPLES;
        final double approximation = TOTAL_AREA * ratio;
        System.out.println( "PI = " + Math.PI );
        System.out.println( "PI ~ " + approximation );
    }
}
```

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
The Area of a Circle
Implementation
For Friday
Acknowledgements
References
About this Document

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
The Area of a Circle
Implementation
For Friday
Acknowledgements
References
About this Document

Implementation

Java

```
import java.util.Random;
public class MonteCarlo {
    // Total number of random points.
    private static final int TOTAL_SAMPLES = 1000000;
    // Radius of base circle.
    private static final double RADIUS = 1.0;
    // Total area of extended square.
    private static final double TOTAL_AREA = 4.0 * RADIUS;

    public static void main( String[] args ) {
        final Random rand = new Random( );
        // Initialise number of random points inside circle.
        int hits = 0;
        for ( int sample = TOTAL_SAMPLES; sample-- != 0; ) {
            // Generate next random point.
            final double x = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            final double y = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            // Increment hits if point is inside circle.
            hits += x*x + y*y <= RADIUS * RADIUS ? 1 : 0;
        }
        // Compute approximation of area of circle.
        final double ratio = (double)hits / TOTAL_SAMPLES;
        final double approximation = TOTAL_AREA * ratio;
        System.out.println( "PI = " + Math.PI );
        System.out.println( "PI ˜ " + approximation );
    }
}
```

Implementation

Java

```
import java.util.Random;
public class MonteCarlo {
    // Total number of random points.
    private static final int TOTAL_SAMPLES = 1000000;
    // Radius of base circle.
    private static final double RADIUS = 1.0;
    // Total area of extended square.
    private static final double TOTAL_AREA = 4.0 * RADIUS;

    public static void main( String[] args ) {
        final Random rand = new Random( );
        // Initialise number of random points inside circle.
        int hits = 0;
        for ( int sample = TOTAL_SAMPLES; sample-- != 0; ) {
            // Generate next random point.
            final double x = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            final double y = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            // Increment hits if point is inside circle.
            hits += x*x + y*y <= RADIUS * RADIUS ? 1 : 0;
        }
        // Compute approximation of area of circle.
        final double ratio = (double)hits / TOTAL_SAMPLES;
        final double approximation = TOTAL_AREA * ratio;
        System.out.println( "PI = " + Math.PI );
        System.out.println( "PI ˜ " + approximation );
    }
}
```

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
The Area of a Circle
Implementation
For Friday
Acknowledgements
References
About this Document

Implementation

Java

```
import java.util.Random;
public class MonteCarlo {
    // Total number of random points.
    private static final int TOTAL_SAMPLES = 1000000;
    // Radius of base circle.
    private static final double RADIUS = 1.0;
    // Total area of extended square.
    private static final double TOTAL_AREA = 4.0 * RADIUS;

    public static void main( String[] args ) {
        final Random rand = new Random( );
        // Initialise number of random points inside circle.
        int hits = 0;
        for ( int sample = TOTAL_SAMPLES; sample-- != 0; ) {
            // Generate next random point.
            final double x = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            final double y = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            // Increment hits if point is inside circle.
            hits += x*x + y*y <= RADIUS * RADIUS ? 1 : 0;
        }
        // Compute approximation of area of circle.
        final double ratio = (double)hits / TOTAL_SAMPLES;
        final double approximation = TOTAL_AREA * ratio;
        System.out.println( "PI = " + Math.PI );
        System.out.println( "PI ˜ " + approximation );
    }
}
```

Sentinel Values
break and continue
Common Algorithms
Nested Loops
Simulations
The Area of a Circle
Implementation
For Friday
Acknowledgements
References
About this Document

Implementation

Java

```java
import java.util.Random;
public class MonteCarlo {
    // Total number of random points.
    private static final int TOTAL_SAMPLES = 1000000;
    // Radius of base circle.
    private static final double RADIUS = 1.0;
    // Total area of extended square.
    private static final double TOTAL_AREA = 4.0 * RADIUS;

    public static void main( String[] args ) {
        final Random rand = new Random( );
        // Initialise number of random points inside circle.
        int hits = 0;
        for ( int sample = TOTAL_SAMPLES; sample-- != 0; ) {
            // Generate next random point.
            final double x = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            final double y = 2.0 * RADIUS * (rand.nextDouble( ) - 0.5);
            // Increment hits if point is inside circle.
            hits += x*x + y*y <= RADIUS * RADIUS ? 1 : 0;
        }
        // Compute approximation of area of circle.
        final double ratio = (double)hits / TOTAL_SAMPLES;
        final double approximation = TOTAL_AREA * ratio;
        System.out.println( "PI = " + Math.PI );
        System.out.println( "PI ~ " + approximation );
    }
}
```

For Friday

- Implement the integer to string conversion method.
  - You should be able to do this from scratch.
- Study Sections 5.5, and 5.7–5.9.

Sentinel Values

break and continue

Common Algorithms

Nested Loops

Simulations

For Friday

Acknowledgements

References

About this Document

Acknowledgements

- This lecture corresponds to [*Big Java, Early Objects*, Sections 5.5, 5.7–5.9].

Bibliography

Horstfmann, Cay S. *Big Java, Early Objects.* International Student Version. Wiley. ISBN: 978-1-118-31877-5.

Software Development

M. R. C. van Dongen

Sentinel Values

break and continue

Common Algorithms

Nested Loops

Simulations

For Friday

Acknowledgements

References

About this Document

About this Document

- This document was created with `pdflatex`.
- The LaTeX document class is `beamer`.

Sentinel Values

`break` and `continue`

Common Algorithms

Nested Loops

Simulations

For Friday

Acknowledgements

References

About this Document