File Edit View Help

File Edit View Help

Programming Review

Part 01

File Edit View Help

Computing Basics

File Edit View Help


File Edit View Help
MAGICAL

File Edit View Help

File Edit View Help

File Edit View Help

INPUT

(Data)

File Edit View Help

INPUT

(Data)

File Edit View Help

INPUT
(Data)

Output
(Information)

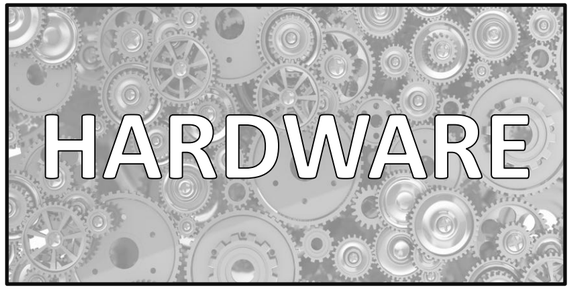
File Edit View Help

File Edit View Help

HARDWARE

SOFTWARE

File Edit View Help



File Edit View Help

HARDWARE

CPU

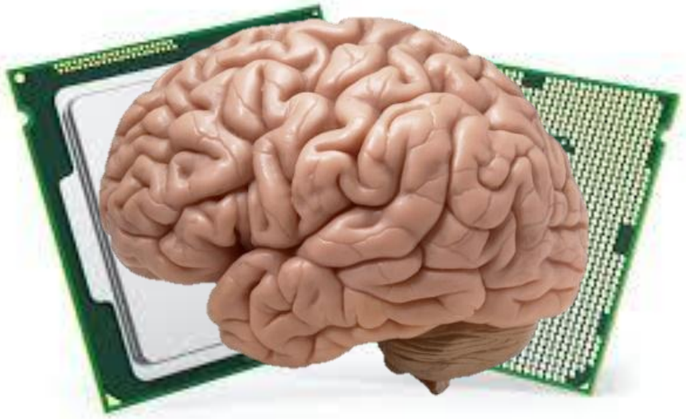
Memory

File Edit View Help

HARDWARE

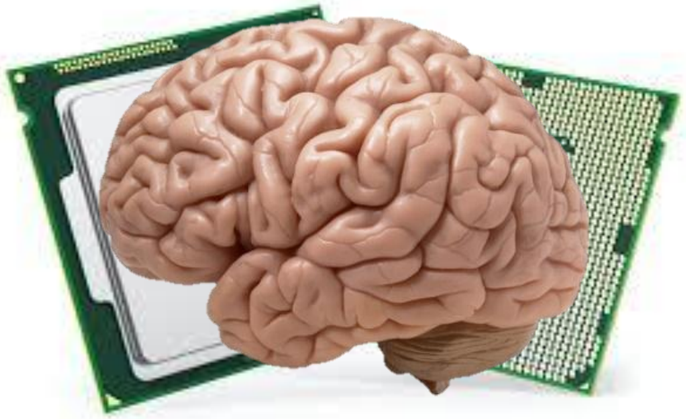
File Edit View Help

HARDWARE



File Edit View Help

HARDWARE



File Edit View Help

HARDWARE

File Edit View Help

HARDWARE

RAM

File Edit View Help

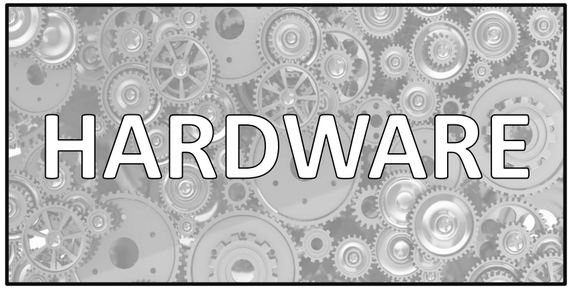
HARDWARE

Secondary

File Edit View Help

HARDWARE

File Edit View Help



Memory	
Address	Value
…	
256	01000001
260	01000010
264	01000010
268	01000001
…	

File Edit View Help

Running Software

Secondary

Main

CPU

File Edit View Help

Running Software

Secondary

Load

Main

CPU

File Edit View Help

Running Software

Secondary

Load

Main

Run Code

CPU

File Edit View Help

Running Software

Secondary

Load

Main

Run Code

Store Info

CPU

File Edit View Help

File Edit View Help

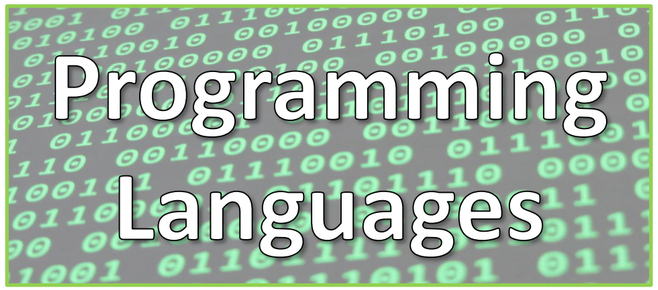
Algorithm
(ˈalgəˌriT_Həm) *noun*
A set of instructions to solve a problem

File Edit View Help

SOFTWARE

Program (prō͵gram) *noun*
A set of instructions for a computer to follow.

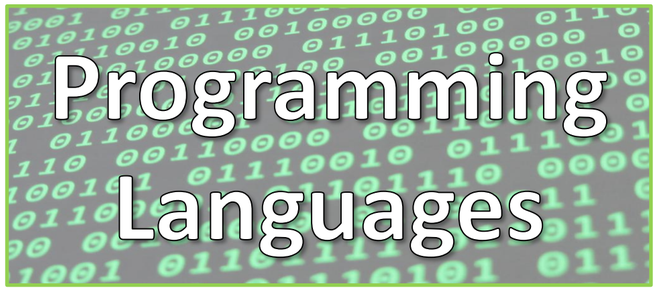
File Edit View Help



File Edit View Help

Programming Languages

File Edit View Help



LOW LEVEL

File Edit View Help

Programming Languages

LOW LEVEL

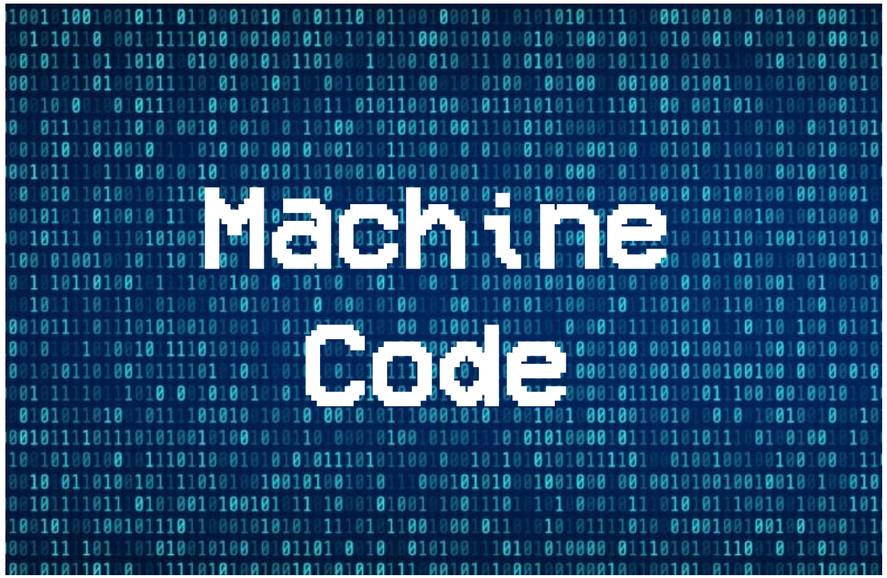
```
*****************************************
* FUNCTION: INCH - Input character
* INPUT: none
* OUTPUT: char in acc A
* DESTROYS: acc A
* CALLS: none
* DESCRIPTION: Gets 1 character from terminal

C010 B6 80 04  INCH    LDA A  ACIA      GET STATUS
C013 47                ASR A            SHIFT RDRF FLAG INTO CARRY
C014 24 FA             BCC    INCH      RECIEVE NOT READY
C016 B6 80 05          LDA A  ACIA+1    GET CHAR
C019 84 7F             AND A  #$7F      MASK PARITY
C01B 7E C0 79          JMP    OUTCH     ECHO & RTS
```

File Edit View Help

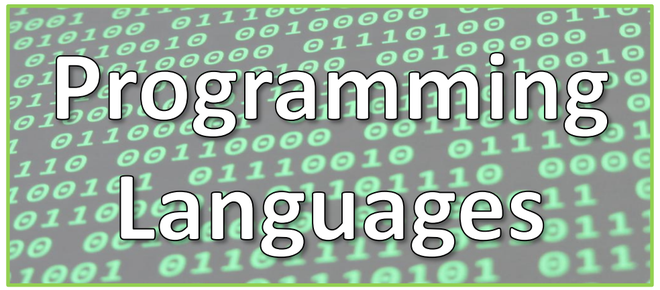
Programming Languages

LOW LEVEL




```
                                    ************************************
                                    * FUNCTION: INCH - Input character
                                    * INPUT: none
                                    * OUTPUT: char in acc A
                                    * DESTROYS: acc A
                                    * CALLS: none
                                    * DESCRIPTION: Gets 1 character from terminal
                                    Assembly
C010 B6 80 04   INCH    LDA A  ACIA     GET STATUS
C013 47                 ASR A           SHIFT RDRF FLAG INTO CARRY
C014 24 FA              BCC    INCH     RECIEVE NOT READY
C016 B6 80 05           LDA A  ACIA+1   GET CHAR
C019 84 7F              AND A  #$7F     MASK PARITY
C01B 7E C0 79           JMP    OUTCH    ECHO & RTS
```

File Edit View Help

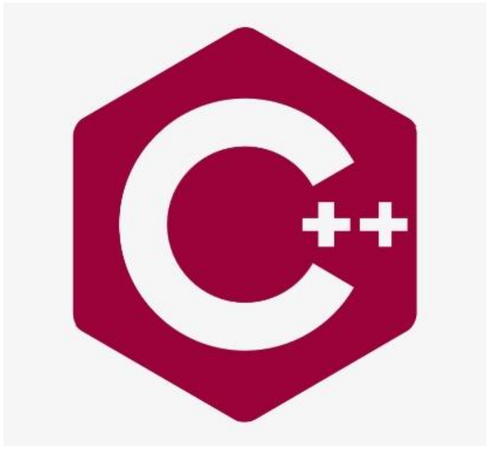


High Level

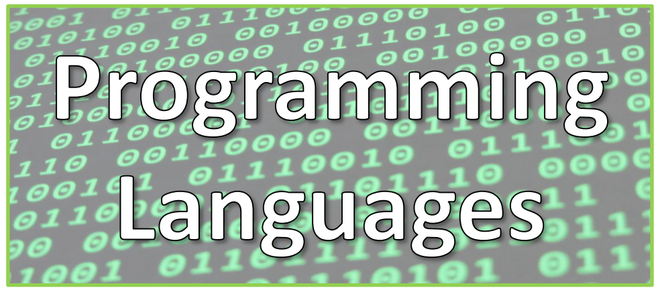
File Edit View Help

Programming Languages

High Level

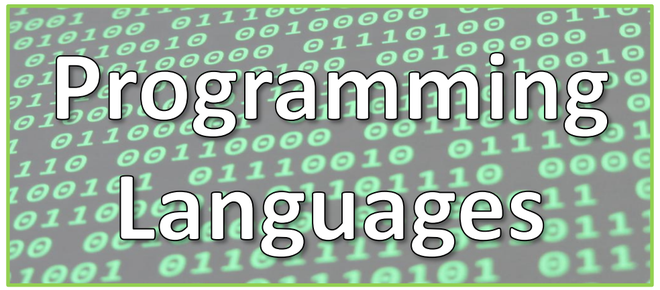


File Edit View Help



High Level

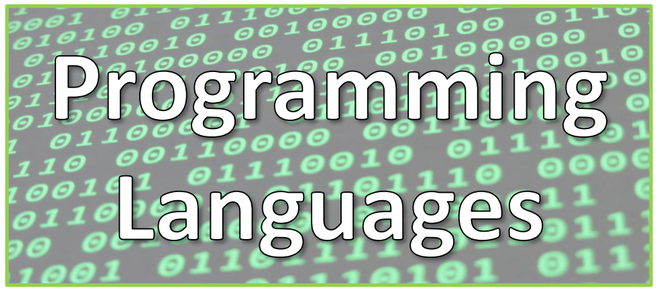
File Edit View Help



High Level

Nouns and Verbs

File Edit View Help



High Level

Syntax

File Edit View Help

Programming Languages

File Edit View Help

Programming Languages

Compiler

File Edit View Help

File Edit View Help

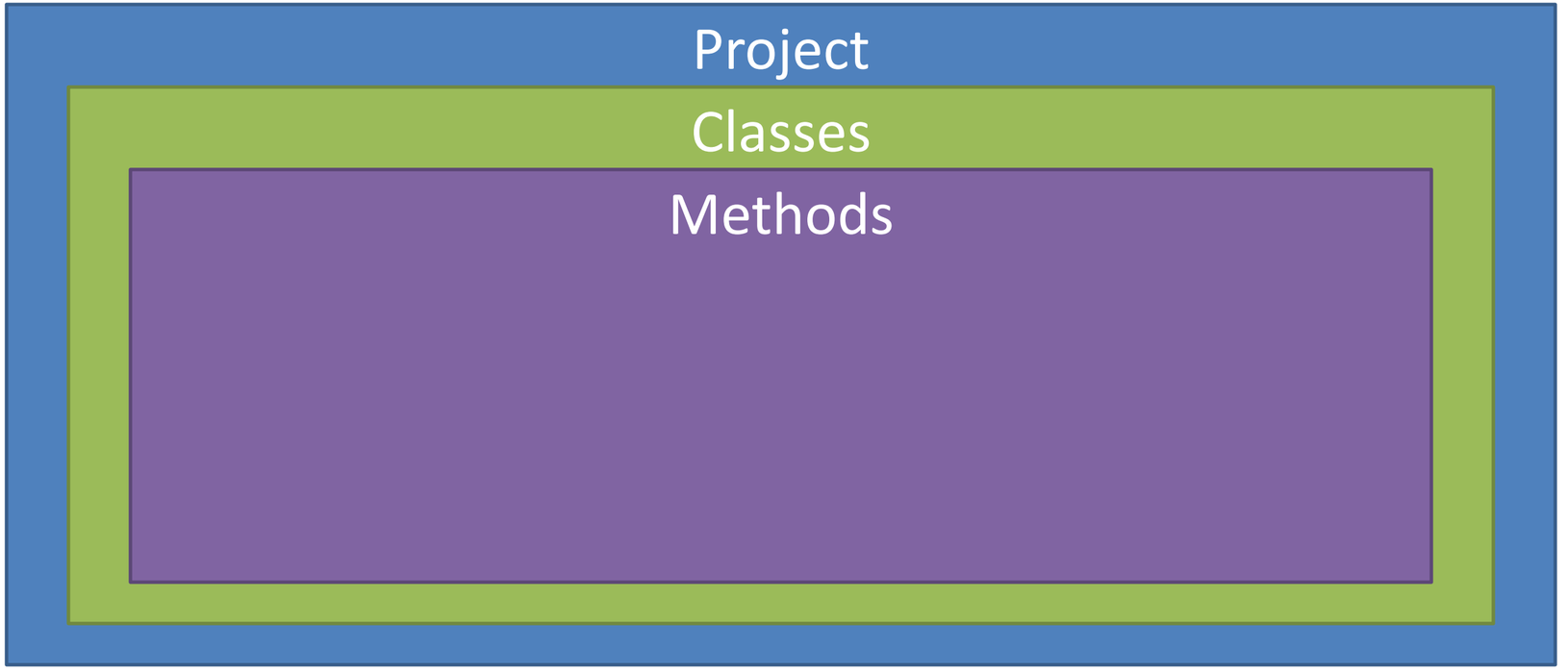
Project

File Edit View Help

Project

Classes

File Edit View Help



File Edit View Help

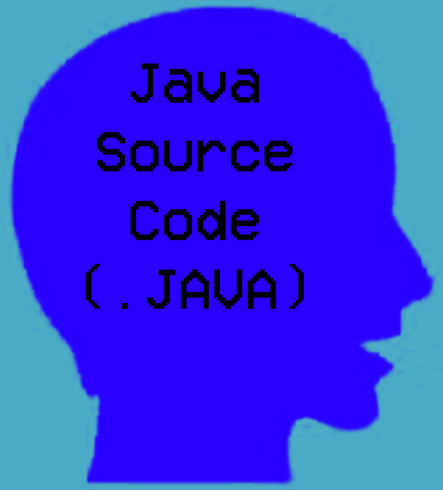
Classes

Methods

- Source Code in files with “.JAVA” extension
- The filename must MATCH the name of the class
- Everything is an “Object”

File Edit View Help

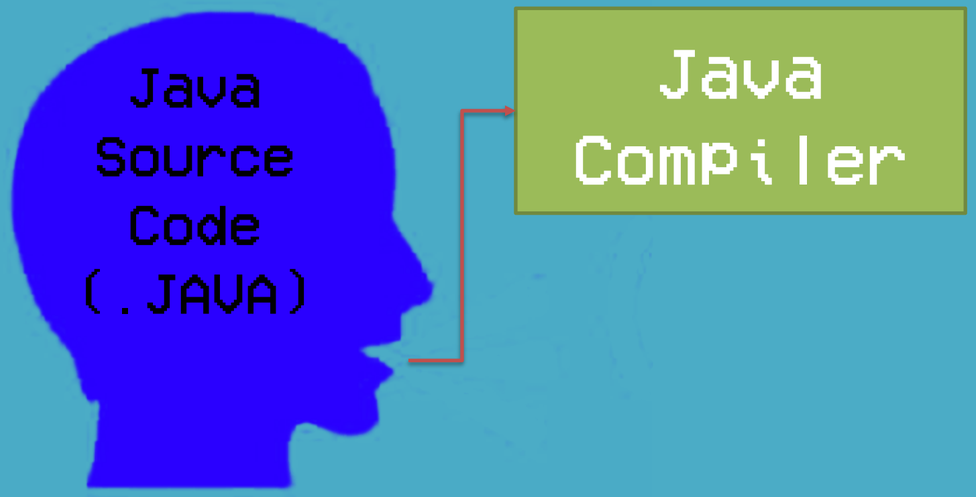
Compilation



Running

File Edit View Help

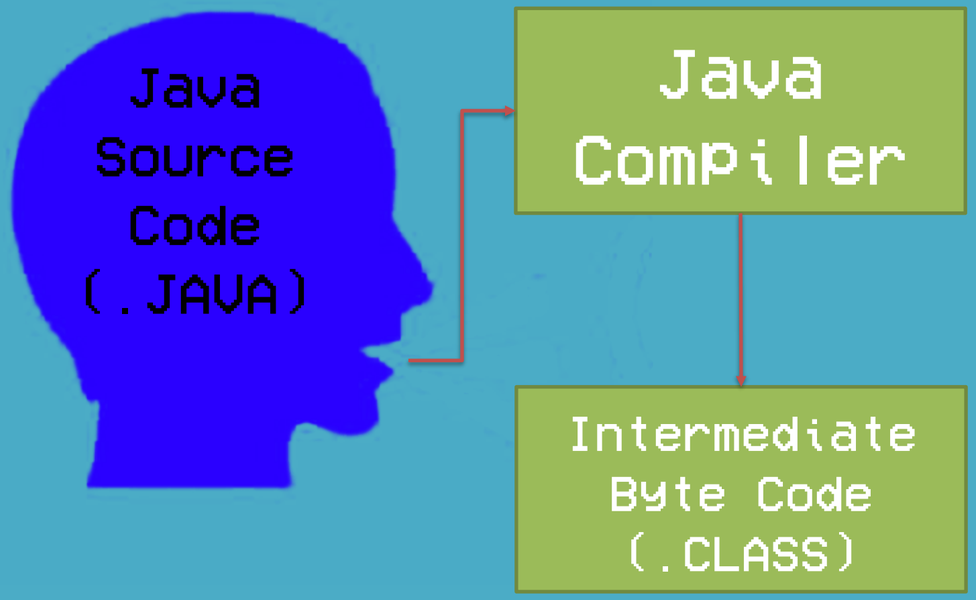
Compilation



Running

File Edit View Help

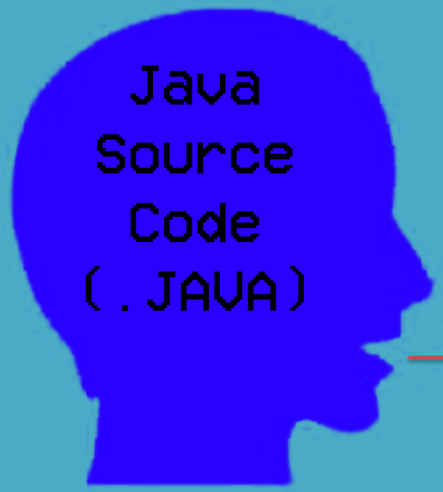
Compilation



Running

File Edit View Help

Compilation



Java Compiler

Intermediate Byte Code (.CLASS)

Running

Java Virtual Machine (JVM)

File Edit View Help

Compilation

Java Source Code (.JAVA)

Java Compiler

Intermediate Byte Code (.CLASS)

Running

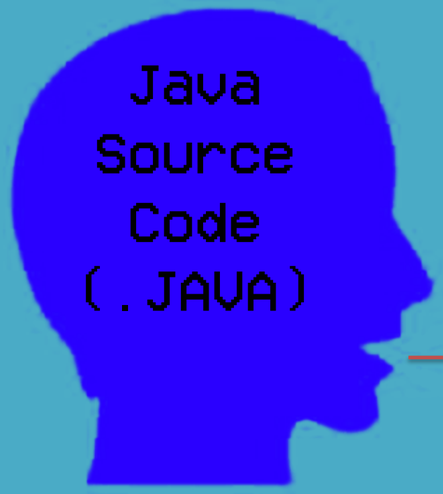
Java Virtual Machine (JVM)

File Edit View Help

Integrated Development Environment (IDE)

- Software for writing and testing Software.
- Not necessary but speeds up development.
- Eclipse, IntelliJ, VSCode, NetBeans, etc.

Compilation



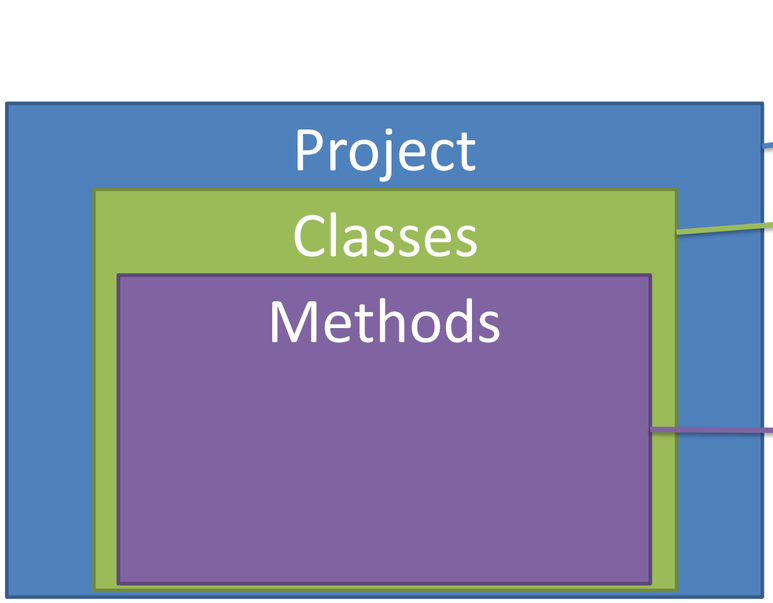
Java Compiler

Intermediate Byte Code (.CLASS)

Running

Java Virtual Machine (JVM)

File Edit View Help



Package Explorer

- CSCE145_HelloWorld
  - JRE System Library
  - src
    - (default package)
      - HelloWorld.java

HelloWorld.java

```
1  /*
2   * Written by Dr. JJ Shepherd
3   * Multi-line comment ignored by the compiler
4   */
5  //Single line comment also ignored by the compiler
6  import java.util.Scanner;//Includes the datatype "Scanner" for input
7  public class HelloWorld //Class
8  {
9      //Inside the Body of the Class
10     public static void main(String[] args) //Method
11     {
12         //Inside the Body of the Method.
13         //The main method is the "entry point" of the software,
14         //or where the computer begins running each statement.
15         System.out.println("Hello World!");//Outputs "Hello World" to the console
16         System.out.println("I can add numbers and such. So... Gimme NUMBERS!!!");
17         //Creates a Scanner for input called "keyboard"
18         Scanner keyboard = new Scanner(System.in);
19         //Two whole number (integer) values are inputted from console via the Scanner
20         //and stored into two variables "value1" and "value2"
21         int value1 = keyboard.nextInt();
22         int value2 = keyboard.nextInt();
23         //Values are added together and stored in another variable called "result"
24         int result = value1 + value2;
25         //The values are each part are outputted to the console.
26          System.out.println("The results of "+value1+" + "+value2+" is "+result);
27     }
28 }
29
```

File Edit View Help

Package Explorer

- CSCE145_HelloWorld
  - JRE System Library
  - src
    - (default package)
      - HelloWorld.java

HelloWorld.java

```
/*
 * Written by Dr. JJ Shepherd
 * Multi-line comment ignored by the compiler
 */
//Single line comment also ignored by the compiler
import java.util.Scanner;//Includes the datatype "Scanner" for input
public class HelloWorld //Class
{
    //Inside the Body of the Class
    public static void main(String[] args) //Method
    {
        //Inside the Body of the Method.
        //The main method is the "entry point" of the software,
        //or where the computer begins running each statement.
        System.out.println("Hello World!");//Outputs "Hello World" to the console
        System.out.println("I can add numbers and such. So... Gimme NUMBERS!!!");
        //Creates a Scanner for input called "keyboard"
        Scanner keyboard = new Scanner(System.in);
        //Two whole number (integer) values are inputted from console via the Scanner
        //and stored into two variables "value1" and "value2"
        int value1 = keyboard.nextInt();
        int value2 = keyboard.nextInt();
        //Values are added together and stored in another variable called "result"
        int result = value1 + value2;
        //The values are each part are outputted to the console.
         System.out.println("The results of "+value1+" + "+value2+" is "+result);
    }
}
```

CSCE145_HolloWorld Project Folder

Name

- .settings
- bin
- src
- .classpath
- .project

CSCE145_HolloWorld src Folder

Name

- HelloWorld.java

Java Compiler

CSCE145_HolloWorld Project Folder

Name

- .settings
- bin
- src
- .classpath
- .project

CSCE145_HolloWorld bin Folder

Name

- HelloWorld.class

Java Virtual Machine (JVM)

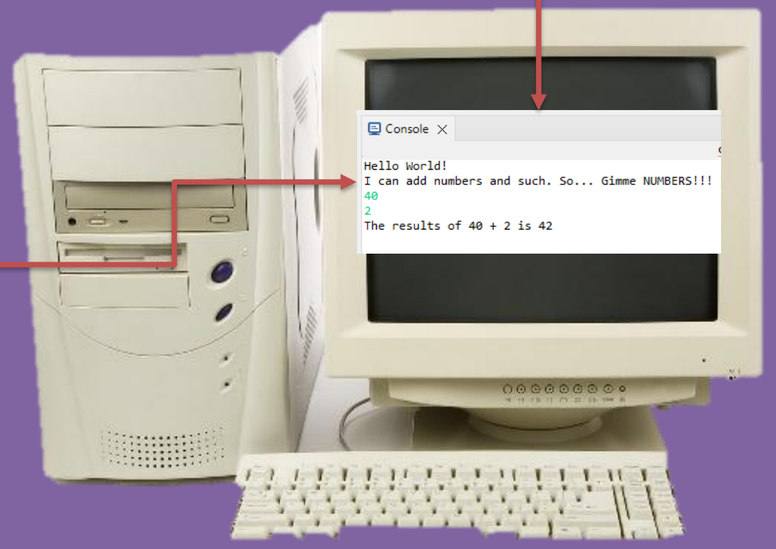
Java Source Code (.JAVA)

Java Compiler

Intermediate Byte Code (.CLASS)

Compilation

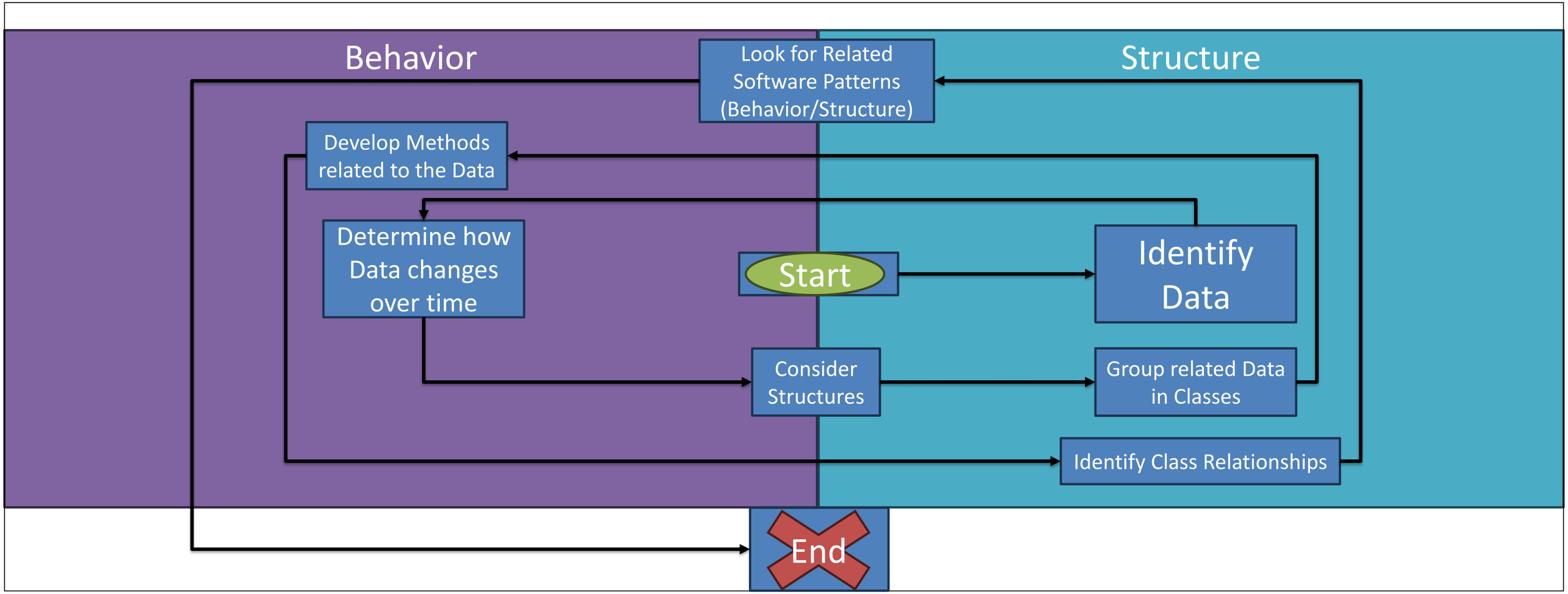
Java Virtual Machine (JVM)



Running

File Edit View Help

Problem Solving



File Edit View Help

HELLO WORLD!


File Edit View Help
HELLO WORLD!


File Edit View Help
HELLO ARRAYS!

Arrays

- A collection (data structure) of items of the same type
- Fixed, contiguous block in memory
- Cannot resize in memory
  - Size of the array needs to be known before it is created
- Java arrays are considered “Objects”
  - Separated reference and contents
  - Have built in properties like “.length”
- When arrays are constructed all items are assumed to be assigned default values, in Java
- Indices (singular “index”) is how we can access and modify values in an array
- Valid indices start from 0 until Length – 1
  - If an array had 10 elements, then the valid indices are from 0 to 9
- Array’s “best friend” is a for-loop
  - The loop can index into the array using its counter

Syntax

```
//Declaring and Constructing an Array
<<type>>[] <<identifier>> = new <<type>>[<<size>>];
//Indexing into an array to access a value
<<identifier>>[<<index>>];
//Indexing into an array to assign / modify a value
<<identifier>>[<<index>>] = <<value>>;
```

Examples

```
int[] i = new int[5];
i[2] = 22;
System.out.println(i[2]);
```

Sorting Algorithms

- Problem:
  - –Given any array of integers, develop an algorithm that sorts the values from smallest to largest.
- Selection Sort
  1. Start from index 0
  2. Assume the starting index has the smallest value and record that index
  3. Sequentially check every other value
  4. If a value is found that is smaller at another index, then record that current index
  5. Once all values have been checked if the recorded index does not match the current index then swap those values
  6. Increase the starting index by 1
  7. Repeat 2 through 6 until the staring index >= length

Example

Index	0	1	2	3	4	5	6	7
Value	10	8	7	6	12	5	11	9

Sorting Algorithms

- Problem:
  - –Given any array of integers, develop an algorithm that sorts the values from smallest to largest.
- Selection Sort
  1. Start from index 0
  2. Assume the starting index has the smallest value and record that index
  3. Sequentially check every other value
  4. If a value is found that is smaller at another index, then record that current index
  5. Once all values have been checked if the recorded index does not match the current index then swap those values
  6. Increase the starting index by 1
  7. Repeat 2 through 6 until the staring index >= length

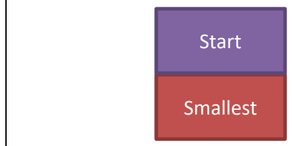
Example

Index	0	1	2	3	4	5	6	7
Value	10	8	7	6	12	5	11	9

Sorting Algorithms

- Problem:
  - –Given any array of integers, develop an algorithm that sorts the values from smallest to largest.
- Selection Sort
  1. Start from index 0
  2. Assume the starting index has the smallest value and record that index
  3. Sequentially check every other value
  4. If a value is found that is smaller at another index, then record that current index
  5. Once all values have been checked if the recorded index does not match the current index then swap those values
  6. Increase the starting index by 1
  7. Repeat 2 through 6 until the staring index >= length

Example

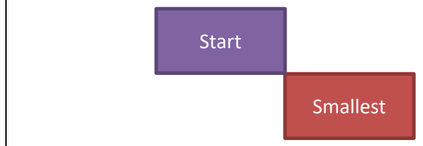


Index	0	1	2	3	4	5	6	7
Value	10	8	7	6	12	5	11	9

Sorting Algorithms

- Problem:
  - –Given any array of integers, develop an algorithm that sorts the values from smallest to largest.
- Selection Sort
  1. Start from index 0
  2. Assume the starting index has the smallest value and record that index
  3. Sequentially check every other value
  4. If a value is found that is smaller at another index, then record that current index
  5. Once all values have been checked if the recorded index does not match the current index then swap those values
  6. Increase the starting index by 1
  7. Repeat 2 through 6 until the staring index >= length

Example

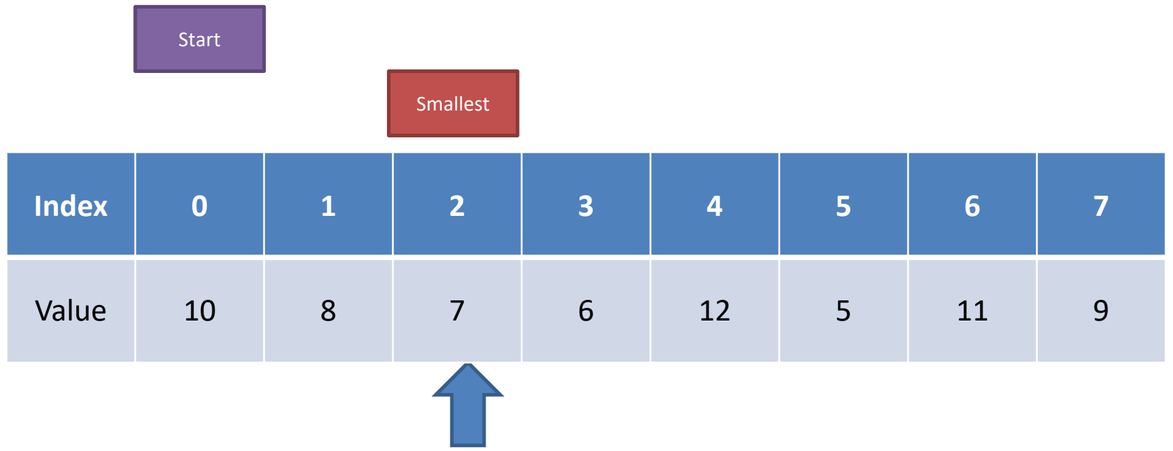


Index	0	1	2	3	4	5	6	7
Value	10	8	7	6	12	5	11	9

Sorting Algorithms

- Problem:
  - –Given any array of integers, develop an algorithm that sorts the values from smallest to largest.
- Selection Sort
  1. Start from index 0
  2. Assume the starting index has the smallest value and record that index
  3. Sequentially check every other value
  4. If a value is found that is smaller at another index, then record that current index
  5. Once all values have been checked if the recorded index does not match the current index then swap those values
  6. Increase the starting index by 1
  7. Repeat 2 through 6 until the staring index >= length

Example

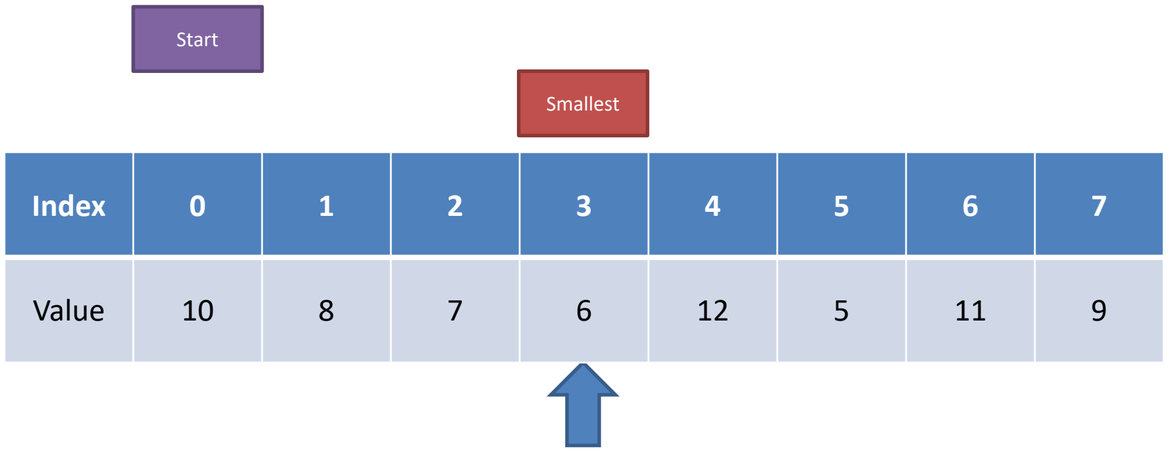


Index	0	1	2	3	4	5	6	7
Value	10	8	7	6	12	5	11	9

Sorting Algorithms

- Problem:
  - –Given any array of integers, develop an algorithm that sorts the values from smallest to largest.
- Selection Sort
  1. Start from index 0
  2. Assume the starting index has the smallest value and record that index
  3. Sequentially check every other value
  4. If a value is found that is smaller at another index, then record that current index
  5. Once all values have been checked if the recorded index does not match the current index then swap those values
  6. Increase the starting index by 1
  7. Repeat 2 through 6 until the staring index >= length

Example

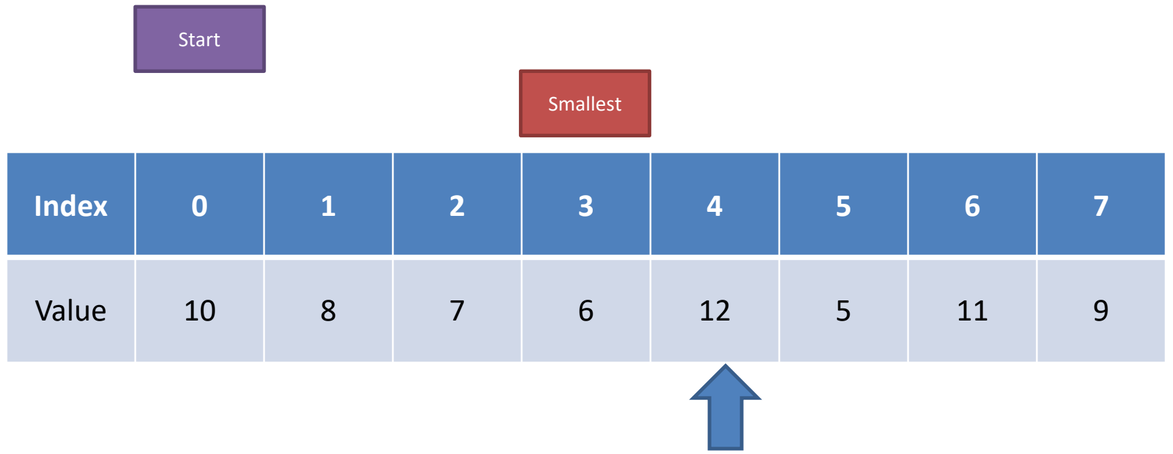


Index	0	1	2	3	4	5	6	7
Value	10	8	7	6	12	5	11	9

Sorting Algorithms

- Problem:
  - –Given any array of integers, develop an algorithm that sorts the values from smallest to largest.
- Selection Sort
  1. Start from index 0
  2. Assume the starting index has the smallest value and record that index
  3. Sequentially check every other value
  4. If a value is found that is smaller at another index, then record that current index
  5. Once all values have been checked if the recorded index does not match the current index then swap those values
  6. Increase the starting index by 1
  7. Repeat 2 through 6 until the staring index >= length

Example

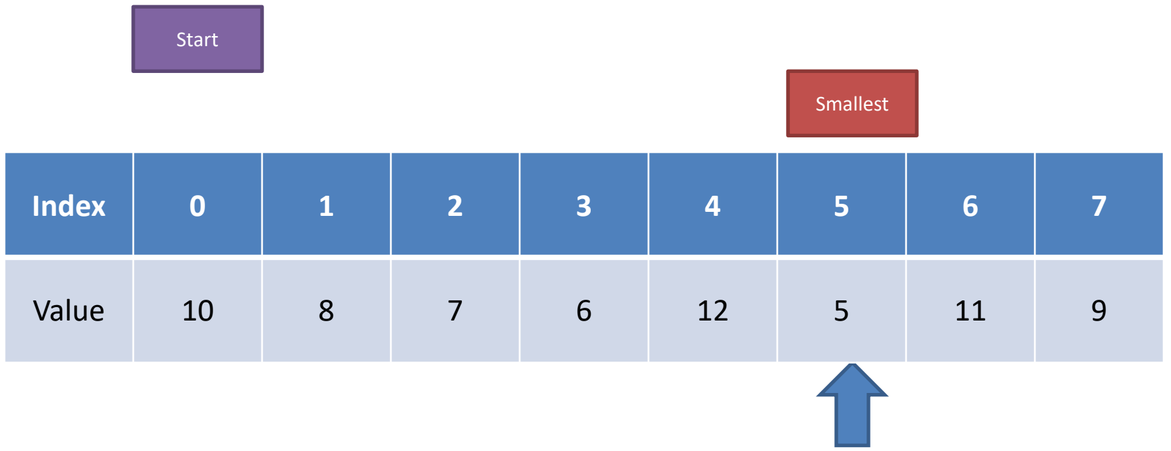


Index	0	1	2	3	4	5	6	7
Value	10	8	7	6	12	5	11	9

Sorting Algorithms

- Problem:
  - –Given any array of integers, develop an algorithm that sorts the values from smallest to largest.
- Selection Sort
  1. Start from index 0
  2. Assume the starting index has the smallest value and record that index
  3. Sequentially check every other value
  4. If a value is found that is smaller at another index, then record that current index
  5. Once all values have been checked if the recorded index does not match the current index then swap those values
  6. Increase the starting index by 1
  7. Repeat 2 through 6 until the staring index >= length

Example

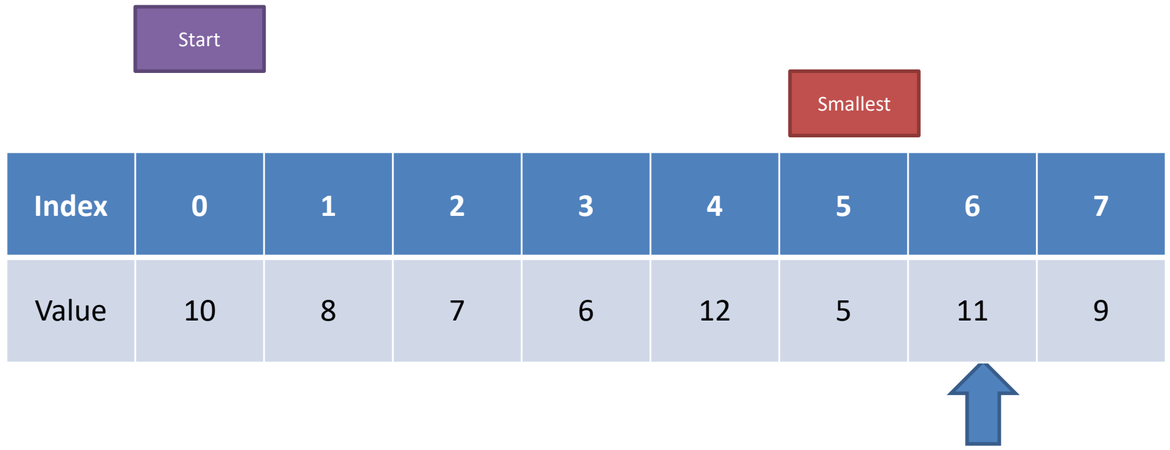


Index	0	1	2	3	4	5	6	7
Value	10	8	7	6	12	5	11	9

Sorting Algorithms

- Problem:
  - –Given any array of integers, develop an algorithm that sorts the values from smallest to largest.
- Selection Sort
  1. Start from index 0
  2. Assume the starting index has the smallest value and record that index
  3. Sequentially check every other value
  4. If a value is found that is smaller at another index, then record that current index
  5. Once all values have been checked if the recorded index does not match the current index then swap those values
  6. Increase the starting index by 1
  7. Repeat 2 through 6 until the staring index >= length

Example

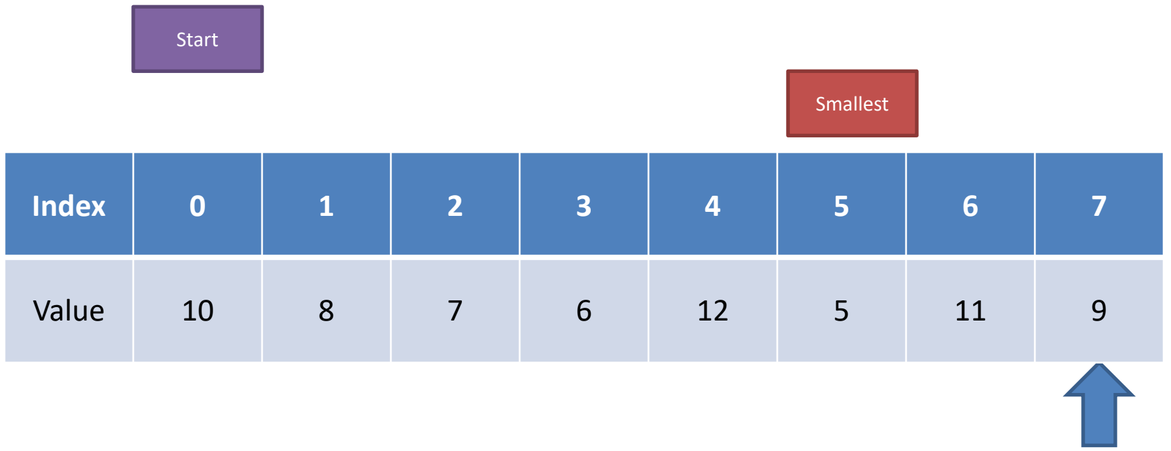


Index	0	1	2	3	4	5	6	7
Value	10	8	7	6	12	5	11	9

Sorting Algorithms

- Problem:
  - –Given any array of integers, develop an algorithm that sorts the values from smallest to largest.
- Selection Sort
  1. Start from index 0
  2. Assume the starting index has the smallest value and record that index
  3. Sequentially check every other value
  4. If a value is found that is smaller at another index, then record that current index
  5. Once all values have been checked if the recorded index does not match the current index then swap those values
  6. Increase the starting index by 1
  7. Repeat 2 through 6 until the staring index >= length

Example



Index	0	1	2	3	4	5	6	7
Value	10	8	7	6	12	5	11	9

Sorting Algorithms

- Problem:
  - –Given any array of integers, develop an algorithm that sorts the values from smallest to largest.
- Selection Sort
  1. Start from index 0
  2. Assume the starting index has the smallest value and record that index
  3. Sequentially check every other value
  4. If a value is found that is smaller at another index, then record that current index
  5. Once all values have been checked if the recorded index does not match the current index then swap those values
  6. Increase the starting index by 1
  7. Repeat 2 through 6 until the staring index >= length

Example

Start

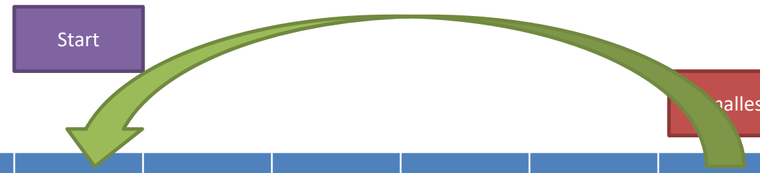
Smallest

Index	0	1	2	3	4	5	6	7
Value	10	8	7	6	12	5	11	9

Sorting Algorithms

- Problem:
  - –Given any array of integers, develop an algorithm that sorts the values from smallest to largest.
- Selection Sort
  1. Start from index 0
  2. Assume the starting index has the smallest value and record that index
  3. Sequentially check every other value
  4. If a value is found that is smaller at another index, then record that current index
  5. Once all values have been checked if the recorded index does not match the current index then swap those values
  6. Increase the starting index by 1
  7. Repeat 2 through 6 until the staring index >= length

Example

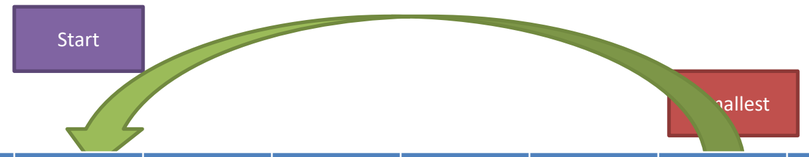


Index	0	1	2	3	4	5	6	7
Value	10	8	7	6	12	5	11	9

Sorting Algorithms

- Problem:
  - –Given any array of integers, develop an algorithm that sorts the values from smallest to largest.
- Selection Sort
  1. Start from index 0
  2. Assume the starting index has the smallest value and record that index
  3. Sequentially check every other value
  4. If a value is found that is smaller at another index, then record that current index
  5. Once all values have been checked if the recorded index does not match the current index then swap those values
  6. Increase the starting index by 1
  7. Repeat 2 through 6 until the staring index >= length

Example



Index	0	1	2	3	4	5	6	7
Value	5	8	7	6	12	10	11	9

Sorting Algorithms

- Problem:
  - –Given any array of integers, develop an algorithm that sorts the values from smallest to largest.
- Selection Sort
  1. Start from index 0
  2. Assume the starting index has the smallest value and record that index
  3. Sequentially check every other value
  4. If a value is found that is smaller at another index, then record that current index
  5. Once all values have been checked if the recorded index does not match the current index then swap those values
  6. Increase the starting index by 1
  7. Repeat 2 through 6 until the staring index >= length

Example

Start

Smallest

Index	0	1	2	3	4	5	6	7
Value	5	8	7	6	12	10	11	9

Sorting Algorithms

- Problem:
  - –Given any array of integers, develop an algorithm that sorts the values from smallest to largest.
- Selection Sort
  1. Start from index 0
  2. Assume the starting index has the smallest value and record that index
  3. Sequentially check every other value
  4. If a value is found that is smaller at another index, then record that current index
  5. Once all values have been checked if the recorded index does not match the current index then swap those values
  6. Increase the starting index by 1
  7. Repeat 2 through 6 until the staring index >= length

Example

Index	0	1	2	3	4	5	6	7
Value	5	8	7	6	12	10	11	9

Sorting Algorithms

- Problem:
  - –Given any array of integers, develop an algorithm that sorts the values from smallest to largest.
- Selection Sort
  1. Start from index 0
  2. Assume the starting index has the smallest value and record that index
  3. Sequentially check every other value
  4. If a value is found that is smaller at another index, then record that current index
  5. Once all values have been checked if the recorded index does not match the current index then swap those values
  6. Increase the starting index by 1
  7. Repeat 2 through 6 until the staring index >= length

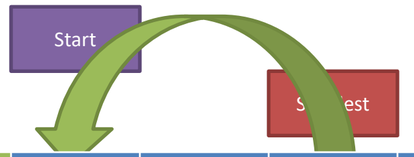
Example

Index	0	1	2	3	4	5	6	7
Value	5	8	7	6	12	10	11	9

Sorting Algorithms

- Problem:
  - –Given any array of integers, develop an algorithm that sorts the values from smallest to largest.
- Selection Sort
  1. Start from index 0
  2. Assume the starting index has the smallest value and record that index
  3. Sequentially check every other value
  4. If a value is found that is smaller at another index, then record that current index
  5. Once all values have been checked if the recorded index does not match the current index then swap those values
  6. Increase the starting index by 1
  7. Repeat 2 through 6 until the staring index >= length

Example



Index	0	1	2	3	4	5	6	7
Value	5	8	7	6	12	10	11	9

Sorting Algorithms

- Problem:
  - –Given any array of integers, develop an algorithm that sorts the values from smallest to largest.
- Selection Sort
  1. Start from index 0
  2. Assume the starting index has the smallest value and record that index
  3. Sequentially check every other value
  4. If a value is found that is smaller at another index, then record that current index
  5. Once all values have been checked if the recorded index does not match the current index then swap those values
  6. Increase the starting index by 1
  7. Repeat 2 through 6 until the staring index >= length

Example

Index	0	1	2	3	4	5	6	7
Value	5	6	7	8	12	10	11	9

Sorting Algorithms

- Problem:
  - –Given any array of integers, develop an algorithm that sorts the values from smallest to largest.
- Selection Sort
  1. Start from index 0
  2. Assume the starting index has the smallest value and record that index
  3. Sequentially check every other value
  4. If a value is found that is smaller at another index, then record that current index
  5. Once all values have been checked if the recorded index does not match the current index then swap those values
  6. Increase the starting index by 1
  7. Repeat 2 through 6 until the staring index >= length

Example

Index	0	1	2	3	4	5	6	7
Value	5	6	7	8	12	10	11	9

File Edit View Help

A Few Swaps Later

Sorting Algorithms

- Problem:
  - –Given any array of integers, develop an algorithm that sorts the values from smallest to largest.
- Selection Sort
  1. Start from index 0
  2. Assume the starting index has the smallest value and record that index
  3. Sequentially check every other value
  4. If a value is found that is smaller at another index, then record that current index
  5. Once all values have been checked if the recorded index does not match the current index then swap those values
  6. Increase the starting index by 1
  7. Repeat 2 through 6 until the staring index >= length

Example

Index	0	1	2	3	4	5	6	7
Value	5	6	7	8	9	10	11	12

File Edit View Help

Selection Sort Implementation

File Edit View Help

Programming Review

Part 01