|  |  |
| --- | --- |
| **Date Assigned: 9/22/15** | **Date Due: 9/24/15** |
| **Unit:** Methodology | **Turn In List:** **1. Terms, 2. Post timeline, and 3. Grid** |
| *“I can create and use many data types in a simple computer program.”* | |

**Data Types and Variables: A look at the major data types for modern languages**

**Content Objectives:** Students will be able to declare, initialize and assign variable for a program.

|  |
| --- |
| **Starter Activity** |
| void setup() {  size(210, 200);  }  void draw() {  background(255);  stroke(0);  line(0, 150, width, 150);  stroke(255, 0, 0);  strokeWeight(3);  point(mouseX, 150);  fill(0, 0, 255);  displayGrade(mouseX/2);  }  void displayGrade(float grade) {    if (grade>=94) {  text("Assign letter grade A." + grade, 50, 50);  } else if (grade<94 && grade>=90) {  text("Assign letter grade A-." + grade, 50, 50);  } else if (grade<89 && grade>=86) {  text("Assign letter grade B+." + grade, 50, 50);  } else if (grade<86 && grade>=82) {  text("Assign letter grade B." + grade, 50, 50);  } else if (grade<82 && grade>=79) {  text("Assign letter grade B-." + grade, 50, 50);  } else if (grade<79 && grade>=76) {  text("Assign letter grade C+." + grade, 50, 50);  } else if (grade<76 && grade>=73) {  text("Assign letter grade C." + grade, 50, 50);  } else if (grade<73 && grade>=69) {  text("Assign letter grade C-." + grade, 50, 50);  } else if (grade<69 && grade>=66) {  text("Assign letter grade D+" + grade, 50, 50);  } else if (grade<66 && grade>=63) {  text("Assign letter grade D." + grade, 50, 50);  } else if (grade<63 && grade>=61) {  text("Assign letter grade D-." + grade, 50, 50);  } else  text("Assign letter grade F." + grade, 50, 50);  } |

|  |  |
| --- | --- |
| **Key Terms:** | |
| Interpreted Language | An interpreted language is a programming language that most implentations execute the code directly |
| Compiled Language | A compiled language is a programming language that’s implentations is typically compilers and not interpreters |
| Low Level Language | Amount of abstraction, closer to machine, looking at interaction between software and hardware. |
| High Level Language | Ultimately, programs written in a **high**-**level language** must be translated into machine **language** by a compiler or interpreter. |
| Execute | The execution line in the code |
| Identifiers | What identifies in the code |
| Declare Variables | Name of any data type/variable that you declare |
| Initialize Variables | Name of any data type/variable that you initialize in the code |
| Assign Variables | Name of any data type/variable that you assign a job in the code |

|  |
| --- |
| **Assignment:** |
| For each data type give the following information. Use the Processing reference as an aid (note that all data types follow the java standard.) You may write N/A where applicable.   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  | **Memory Used** | **Possible Values (Min)** | **Possible Values (Max)** | **Purpose** | **Syntax** | | boolean | 1 byte | false | true | Determines whether or not the function is true or false | Boolean Var  Boolean Var = booleanvalue | | byte | 1 byte | -128 | 127 | Convenient data types for sending information | Byte var  Byte var = value | | char | 2/16 bytes/bits | N/A | N/A | Data type for characters | char var  char var = value | | color | 32 bit | 0 | 265 | Datatype for storing color values | Color (0,0,256) | | double | 64 bit | -3.40282347E+38 | 3.40282347E+38 | Datatype for floating-point numbers larger than those that can be stored in a float | double var  double var = value | | float | 32 bit | -3.40282347E+38 | 3.40282347E+38 | Data type for floating-point numbers, e.g. numbers that have a decimal point. | float var  float var = value | | int | 32 bit | -2,147,483,648 | 2,147,483,647 | Datatype for integers, numbers without a decimal point. | int var  int var = value | | long | 64 bit | -9,223,372,036,854,775,808 | 9,223,372,036,854,775,807 | Datatype for large integers. | long var  long var = value | | String | 1 byte per character | n/a | n/a | A string is a sequence of characters. | String(data)  String(data, offset, length) | | XML | File size dependent | n/a | n/a | XML is a representation of an XML object, able to parse XML code. | XML xml; | | Array | Dependent on datatype | Each array has a variable length | Each array has a variable length | An array is a list of data | datatype[] var  var[element] = value  var.length | | ArrayList | Dependent on datatype | n/a | n/a | An ArrayList stores a variable number of objects | ArrayList<Type>()  ArrayList<Type>(initialCapacity) | | Table | File size | n/a | n/a | Table objects store data with multiple rows and columns, much like in a traditional spreadsheet. | Table()  Table(rows) |   Create a new processing project with a medium gray canvas size of 1000 x 1000 pixels and draw a black grid on the first made up of lines at every 100 pixels vertically and horizontally. Provide text labels (100, 200, etc.) on the left margin and top margin. |

Notes (Points of interest, mistakes, lessons learned, web resources, and thoughts):

|  |
| --- |
|  |