ITP 120 (D01A)

Java Programming I

Course Syllabus

Instructor	Jeffrey Elkner
Session	Fall 2020
Meeting Days	B Day
Time	11:50 am - 1:10 pm
Location	Online / Arlington Career Center Room 508
Contact	jde232@email.vccs.edu / jeffrey.elkner@apsus.va

Course Description:

Entails instruction in fundamentals of object-oriented programming using Java. Emphasizes program construction, algorithm development, coding, debugging, and documentation of console and graphical user interface applications. 4 Credits.

General Course Purpose:

This course provides a comprehensive foundation sufficient for a student to write Java programs from scratch in order to meet the minimum programming goals of students who plan to transfer and students who take the course for employment purposes.

Course Objectives:

Upon completion of this course, the student will be able to:

- Design, develop, code, and test Java Programs console applications
- Use primitive data types and flow control statements that are the building blocks of all programming
- Use primitive data types and flow control statements that are the building blocks of all applied appropriately in a Java Program as a solution to a specific problem statement

Major Topics to be Included

- Primitive data types
- Input and Output methods and techniques
- Selection Statements and Looping
- Methods
- Arrays
- Classes and Objects
- Strings
- Files and Text I/O
- Inheritance and Polymorphism

ITP 120 (D01A)

Java Programming I

Course Syllabus

Student Learning Outcomes

Primitive Data Types

- Write Java statements to declare and initialize integer, long, float, double, char, and boolean variables.
- Write Java statements to assign values to variables using expressions containing mathematical and boolean operators.
- Write Java statements to create arithmetic expressions following order of precedence rules.
- Write Java statements using boolean operators: and (&&), or (||), not (!), exclusive or (^).
- Write Java statements using the relational operators to include greater than, greater than or equal to, less than, less than or equal to, is equal to, and not equal to.
- Write Java statements using the arithmetic operators to include add, subtract, multiply, divide, and modulus.
- Write Java statements using the increment and decrement operators.
- Write Java statements using pre and post increment and decrement operators.
- Write Java statements using the short cut arithmetic assignment operators to include +=, -=, *=, /=, and %=.
- Write Java statements incorporating cast operators for primitive data types in arithmetic expressions appropriately.
- Write Java statements to declare named constants.
- Write Java statements using both named and literal constants in java programming statements.

Input and Output

- Console Input
  - Write Java statements to complete console input using Scanner class.
  - Write Java statements including appropriate prompt messages for keyboard input.
- Console Output
  - Write Java statements including print, println, and printf for output statements to the console.
  - Write Java statements for console output including text labels and data values from the program.
  - Write Java statements to include column headings and section headings in program output reports.
- GUI Input
  - Write Java statements to complete GUI input using JOptionPane and showInputDialog.
  - Write Java statements to include a prompt message and a window title for the GUI input.
- GUI Output
  - Write Java statements to complete GUI output using JOptionPane and showMessageDialog.
  - Write Java statements including an output message and window title for the GUI output.
- Write Java statements to validate input of data from the keyboard.

ITP 120 (D01A)
Java Programming I
Course Syllabus

Selection Statements

- Write Java statements to create and use an if statement without an else statement (one branch).
- Write Java statements to create and use an if/else statement (two branches).
- Write Java statements to create and use an if/elseif/else statement (multi-level, many branches).
- Write Java statements to create and use an if statement within the body of another if statement (nested).
- Write Java statements to create and use switch/case statements including default option.
- Write Java statements using the conditional operator.

Loop Statements

- Write Java statements to construct and use a while loop.
- Write Java statements to construct and use a do-while loop.
- Write Java statements to construct and use a for loop.
- Write Java statements to use loop statements appropriately for the following situations:
    - Be able to create and use a counter control loop appropriately.
    - Be able to create and use a data validation loop appropriately.
    - Be able to create a loop that is controlled with a sentinel value.

Methods

- Write Java statements to create a method that takes one or more input arguments.
- Write Java statements to create a method that takes no input arguments.
- Write Java statements to create a method that returns a value.
- Write Java statements to create a method that does not return a value.
- Be able to differentiate between pass by value and pass by reference.
- Describe the difference between static methods and non-static methods.
- Write Java statements to create a method that takes an array as an input argument.
- Write Java statements to create a method that returns an array.
- Write Java statements to create overloaded methods.
- Write Java statements to use local variables in a method.
- Write Java statements which use common methods in the Math class: Math.sqrt, Math, abs, Math.max, Math.min, Math.random, Math.pow.

Arrays

- Write Java statements to declare, create, and initialize a one dimensional array.
- Describe the memory allocation for a one dimensional array including both the array reference and the memory for the array elements.
- Write Java statements to fill an array with values.
- Write Java statements to print out the contents of an array.
- Write Java statements to copy one array to another array.
- Write Java statements to access every element in an array for analysis for the data.

ITP 120 (D01A)

Java Programming I

Course Syllabus

- Describe the concept of parallel arrays and their relationship to classes and objects Write Java statements to declare, create, and initialize multi-dimensional arrays.
- Be able to pass arrays to methods.

Classes and Objects

- Be able to define the relationship between a class and an object.
- Write Java statements to create a class that includes attributes and methods.
- Write Java statements to create objects.
- Be able to write visibility modifiers to include: public, private, protected, and no modifier.
- Write Java statements to create and use static attributes.
- Write Java statements to create and use constant attributes.
- Be able to define the purpose of a constructor.
- Write Java statements to create and use constructors.
- Write Java statements to create and use an overloaded constructor.
- Be able to define the purpose of an accessor method (get method).
- Write Java statements to create appropriate accessor methods for a class (get methods).
- Be able to define the purpose of a mutator method (set method).
- Write Java statements to create appropriate mutator methods for a class (set methods).
- Write Java statements to print out all attributes in a class.
- Be able to describe the following object oriented programming terms:
    - Encapsulation including information hiding and implementation hiding
    - Polymorphism
    - Inheritance
- Write Java statements that pass objects to methods.
- Write Java statements that include objects as attributes.
- Write Java statements that return an object from a method.
- Use classes and objects in programs that you write from scratch.

Strings

- Write Java statements to declare, create, and use a String object.
- Write Java statements using the following methods from the String class:
    - equals, equalsIgnoreCase
    - compareTo, toCharArray()
    - toUpperCase, toLowerCase
    - valueOf, charAt()
    - concat
    - length
    - indexOf and lastIndexOf
    - substring
- Write Java statements to declare, create, and use a Character object.
- Write Java statements using the following methods from the Character class:
    - compareTo
    - equals
- Write Java statements to declare, create, and use a StringBuffer object.

ITP 120 (D01A)

Java Programming I

Course Syllabus

- Write Java statements using the toString from the StringBuffer class.
- Write Java statements to create a toString method for a class that you have created.

Files

- Write Java statements to open a text file for reading.
- Write Java statements to read data from a text file.
- Write Java statements to close a text file.
- Write Java statements to open a text file for writing.
- Write Java statements to write data to a text file.
- Write Java statements using exceptions as required for file input and output.

Inheritance

- Be able to use inheritance in programs and classes.
- Be able to describe the relationship between a super class and a sub class.
- Be able to recognize a sub class from the extends label in a class header.
- Be able to describe the purpose of the super statement when using inheritance.
- Be able to write Java statements for simple classes using inheritance.

Programs

- Be able to write Java programs using the results of an appropriate design methodology as a starting point.
- Be able to write Java programs combining the statements described in this document.
- Be able to write, compile, test, debug, and run Java programs.

ITP 120 (D01A)
Java Programming I
Course Syllabus

Required Time Allocation per Topic

The following table has the breakdown of the time in this course that will be spent on each course topic:

Topic	Hours	Percentage
Primitive data types	6	10%
Input and Output	4	6.67%
Selection Statements	4	6.67%
Loop Statements	4	6.67%
Methods	8	13.33%
Arrays	4	6.67%
Classes and Objects	8	13.33%
Strings	4	6.67%
Files	4	6.67%
Inheritance	4	6.67%
Programs	4	6.67%
Other Optional Topics	2	3.33%
Exams and Quizzes (NOT including final exam)	4	6.67%
Total:	**60**	**100%**

Required Instructional Materials:

Head First Java, 2nd Edition (available through NVCC O'Reilly for Higher Education).
Android Developer Fundamentals Course (available at https://codelabs.developers.google.com/android-training/)

Course Credit: 4 Credits

Policies:

I. Expectations

ITP 120 (D01A)
Java Programming I
Course Syllabus

a. Software Design is a rigorous, college level course that will require sustained and consistent engagement from students.
b. An average of 90 minutes of homework will be assigned for each 90 minutes in class. We will be utilizing a flipped classroom learning environment, where the lecture portion of the course material will be viewed individually at home *before* class meets, and class time will be used for collaborative engagement and discussion.
c. Daily "mini quizzes" at the beginning of class will be used to be sure homework readings and practice have been completed. To be successful in this class, students will be expected to be prepared for these quizzes when they arrive in class.

II. Grading Policies

a. Grading Scale
A= 100 - 90 B= 89 - 80 C= 79 - 70 D=69 - 60 F= 59 and below
b. Students will receive a weekly cumulative letter grade that will incorporate daily quizzes, tests, projects, and presentations. These weekly evaluations can be challenged by the student, *but only during the week immediately following when the evaluation is given*.
c. The average of the weekly evaluations will make up 70% of the final grade, with the course final exam making up 30%.
d. In cases where district grading policies conflict with college grading policies, the high school and college grades may differ; this may include assignment/test retakes, extended assignment due dates, capped minimum grade allowed, among other such district policies.
e. It is important that students check their final NOVA grades in Blackboard as soon as the course(s) completed.

III. Course Policies

a. Academic Integrity

i. The College does not tolerate academic dishonesty. Students who are not honest in their academic work will face disciplinary action along with any grade penalty the instructor imposes. Procedures for disciplinary measures and appeals are outlined in the Student Handbook (http://www.nvcc.edu/students/handbook/). In extreme cases, academic dishonesty may result in dismissal from the College.

ii. Plagiarism: is the act of appropriating passages from the work of another individual, either word for word or in substance, and representing them as one's own work. This includes any submission of written work other than one's own. In short, plagiarism means using the exact words, opinions, or factual information from another person without giving that person credit. Students who are not honest in their academic work will face disciplinary action along with any grade penalty the instructor imposes. For more information about student academic integrity: https://www.nvcc.edu/curcatalog/policies/integrity.html

b. Disabilities

i. Students with disabilities are required to contact NOVA's Office of Disability Support Services (DSS) to discuss possible accommodations. All information is kept confidential and may increase your chances of success in the academic setting. If accommodations are agreed upon, student will receive a Memorandum of Accommodation (MOA) by DSS. For more information about NOVA's DSS office: https://www.nvcc.edu/disability-services.

c. Self-Advocacy

i. Students are expected to reach out to their instructor if they do not understand content or expectations.
ii. College instructors and other college personnel will not talk with a parent without the permission of and presence of the student. The conversation is between the administrator / faculty member and the student. The parent's role is to listen, give moral support, and summarize information and agreements if needed.
iii. Dual enrolled students have access to full NOVA campus services to include tutoring, library, and counseling services; student resources are found here: http://www.nvcc.edu/students/index.html

IV. Course Schedule

a. Critical Course Dates

Course Start Date	Monday, September 14, 2020
Course Drop Date	Friday, October 2, 2020
Course Withdrawal Date	Friday, December 4, 2020
Final Exam Date	Week of January 25 to 29, 2021
Course End Date	Friday, January 29, 2021

b. Final Exam Date: *The final exam will be given during the last week of class, between Monday, January 25 and Friday, January 29.*