**Assignment 5: Object Oriented Design & Programming (with Smalltalk)**

As developers, we look to a variety of approaches to help us understand and analyze problems in order to choose the best methodology for programming the solution. Object-oriented analysis and design is a software development practice which aligns with characteristics found in object oriented programming.

**Object-Oriented Analysis and Design:** Object-oriented analysis and design (OOAD) is a technical approach for analyzing and designing an application by applying an object-oriented paradigm and using visual modeling throughout the development life cycles to enhance product quality.

**Object-Oriented Programming:** Programming by defining objects that send messages to each other. Objects have their own internal (encapsulated) state and public interfaces. Many modern/popular languages support object-oriented programming (Java, C++, Python as examples).

**Smalltalk:** Smalltalk is one of the earliest languages which supports object-oriented programming. Smalltalk is considered "pure" in an OO sense—simply, everything is an object and objects are only communicated with via the sending of messages. The following article is a quick overview of its history, significance in modern programming, and the benefits of learning the fundamentals: [How learning Smalltalk can make you a better developer](https://techbeacon.com/app-dev-testing/how-learning-smalltalk-can-make-you-better-developer)

**Snake Game:** In the snake game, the player controls a dot, square, or other object on a plane. As the object moves forward, it leaves a trail behind it, resembling a moving snake. The player attempts to eat items by running into them. Each item eaten makes the snake longer, so controlling the snake becomes progressively more difficult as the snake’s length increases.

The version of Snake for this assignment is going to be a simple program.

1. The program will run in the terminal
2. Single player
3. The player has to press “enter” every time they want to make a move
4. There is a randomly generated (starting) dot
5. The snake can move up, down, left, or right
6. If the snake eats the dot, it grows and the score goes up by one
7. If the snake runs into the wall or itself, the game resets

**Assignment Overview: For this assignment, you will** use a simple methodology for OOAD analysis and design to code a simple version of the snake game employing basic object oriented programming characteristics:

* Encapsulation
* Abstraction
* Inheritance
* Polymorphism

**Requirements:**

1. Analyze and design the key steps and diagrams for the Snake game using OOAD (see this  [article](http://www.informit.com/articles/article.aspx?p=360440&seqNum=5) as it provides a simplified version of OOAD (with examples) we will use for this assignment) including:

* Defining Use Cases (required: one use case on how people will use the program (must be at least three full sentences in length).
* Defining a (basic) Domain Model (this is a conceptual diagram; shows data and behavior)
* Assign Object Responsibilities and Draw Interaction Diagrams (more defined then the basic domain model)
  + - Identify class names
    - Identify attributes
    - Identify associations

Note: diagrams can be hand drawn though must be clear/readable), other options include an online tool such as <https://www.umlet.com> or smart art in Word)

1. In the Smalltalk language version of your choice, write and test code which implements the snake game (based on your design/analysis)
2. Write a project report (written formally in full sentences) which includes:
   1. An introduction paragraph: introduce the assignment/the problem to be solved; overview of report
   2. Briefly discuss the steps you took and decisions you made to start and solve this assignment (platform, version of Smalltalk, analysis/design, coding…)
   3. Answer: did the analysis and design help you design the code? Briefly explain your answer.
   4. Discuss the use of OOP characteristics in the coding of your snake program.
   5. Concluding paragraph: Concludes the report: wrap up the report (summary of assignment and what you did to meet the requirements; what you learned…).
3. **Deliverables:** You will submit a single Word or PDF document which includes:
   1. The written discussion elements found in #3 (in the above section)
   2. Your design and analysis of the snake game (#1 above)
   3. Video of your program (the working URL to the video should be listed in the report document; at the end of the report). The video must include/show:
      1. Your program compiling
      2. The program running from start to finish showing the following:
         1. Player pressing “enter” every time they want to make a move (showing the moves)
         2. The randomly generated (starting) dot (you must start the game at **least twice** to demonstrate the random start place of this dot)
         3. The snake going up, down, left, or right
         4. If the snake eats the dot, it grows and the score goes up by one
         5. If the snake runs into the wall or itself, the game resetting
   4. List (working URLs are sufficient) all resources used to complete this work (you are expected to explore at least two sources)
   5. Text of your code copied at the end of the report
4. Expectations:
   1. **Compiling errors:** Your solution must compile
   2. **Readability.** Your code should meet basic readability principles:
      1. Separate each component/part with white space.
      2. Align everything in a meaningful way.
   3. **Comments**: you must include enough comments to ensure that the code is described in sufficient detail such that anyone else looking at the code can easily understand the design and the purpose of the code. **For this assignment, your comments should also document where you used object oriented programming characteristics.**
   4. **All code and work associated with this assignment is your own work/written by you**.
   5. **This is not a group project**

**Resources**

**Finding a version of Smalltalk**

List of Smalltalk versions (with links) at the end of this Wikipedia article: w

List of Smalltalk versions (with links & information about each); <https://medium.com/smalltalk-talk/smalltalk-choices-6c39d09a40e6>

**General & help information (Smalltalk)**

Smalltalk, in Sebesta chapter 12

<http://www.smalltalk.org>

<http://ceronio.net/2017/07/first-steps-with-pharo-smalltalk/>

<https://www.codeproject.com/Articles/1241904/Introduction-to-the-Smalltalk-Programming-Language>

<https://squeak.org>

<https://www.whoishostingthis.com/resources/smalltalk/>

<http://books.pharo.org/pharo-by-example/>

<https://techbeacon.com/app-dev-testing/how-learning-smalltalk-can-make-you-better-developer>

<https://medium.com/smalltalk-talk/behold-pharo-the-modern-smalltalk-38e132c46053>

<http://wiki.c2.com/?IfSmalltalkIsSoGoodWhyDoesNobodyUseIt>

**Object Oriented Analysis & Design**

<https://www.codeproject.com/Articles/1137299/Object-Oriented-Analysis-and-Design>

<https://www.cs.colorado.edu/~kena/classes/5828/s07/lectures/20/lecture20.pdf>