Exam 98-380: Introduction to Programming using Block-Based Languages

Candidates for this exam should understand algorithmic flow and be able to describe computer programs, use and implement common program control structures, and describe what the code does in block-based programming languages such as the Touch Develop environment from Microsoft and MIT Scratch. Candidates should also be familiar with the concepts and technologies described here by taking relevant training courses, such as Creative Coding Through Games and Apps (CCGA) or Scratch or Blockly courses. Candidates are expected to have some hands-on experience designing, creating, and publishing code within a block-based programming language.

Microsoft
Technology Associate

Objective Domain

Apply Strategies to Solve Computational Problems

- **Identify basic algorithmic steps to solve simple problems.** Decompose simple problems into steps; sequence processes in the appropriate order; describe storyboards; resolve challenges and errors related to logic or pseudocode
- **Decompose a computational problem into sub-problems.** Describe computer programs that use logical subdivisions; describe solutions that use programmable strategies such as objects, functions, and parameters in the pseudo code provided; identify situations when code can be reviewed

Design Algorithms

- **Create algorithms.** Differentiate problems as easy or hard for computers to solve; apply the concept of iteration; create simple algorithms
- **Analyze game play to identify the algorithmic sequences.** Analyze a game and create a sequence of instructions for playing it; identify an event; create the code for an event in block-based editors; explain the “on every frame” code and event handlers
- **Create and analyze algorithms that can be used to implement animation and movement in code.** Describe animation that uses a series of individual frames; resolve errors in algorithms; create algorithms that can be translated into pseudocode or block-based code; use code to command items on the screen or device
- **Explain sequence, selection, and iteration.** Define loops; identify the control variable; predict the output of loop, random number, and control variable constructs; identify conditional statements; choose the appropriate Boolean logic for specific results

Exam 98-380: Introduction to Programming using Block-Based Languages

Work with Data Representation in Block-Based Programming Languages

- **Represent data in a variety of ways, including text, sounds, pictures, and numbers.** Create code to add and position objects, such as sprites on a screen or device; identify data examples as text, sound, pictures, or numbers; change the parameters of "set frame grid" to work with different sprite sheets; explain the role of cloud variables; explain the impact of variable scope, including cloud variables, global variables, and local or temporary variables; design, create, and populate a table or two-dimensional array; describe multiple uses for data.
- **Employ simple data structures to solve computational problems.** Declare and use variables in a program; use input and variables to calculate new information; describe arrays, lists, and collections; explain the differences between variables and arrays
- **Describe how various types of data are accessed in apps and games .** This objective may include but is not limited to: Name your tenant; set up your first administrator; determine tenant location.

Solve Computational Problems by Using Modeling and Simulation

- **Solve computational problems by using computer and non-computer methods, including unplugged and physical manipulatives.** Use algorithms and Boolean logic; use games and apps to simulate practical tasks such as converting currencies.
- **Represent events typically observed in the natural, physical world by coding simulation and modeling programs.** Create programs and apps that mimic random occurrences; create programs and apps that demonstrate fundamentals of physics such as gravity, acceleration, and bounce.

Code Programs in Block-Based Programming

- **Implement solutions using code.** Identify the basic coding elements of the programming environment; create code for conditional statements; create conditional statements using and, or, and not; create loops; use counting variables
- **Use libraries and built-in functions to facilitate programming solutions.** Code by using ready-made functions related to objects including the game board, wall, obstacle, sprite, string, textbox, collection, and turtle.
- **Make connections between elements of mathematics and computer science.** Plot points on a gaming coordinate system; analyze a game to discover how random numbers are used; use the random range function to generate random numbers; evaluate random numbers by using conditionals

Assess Personal Security in Internet Communications

- **Explain the basic components of Internet communication.** Explain how information travels across the Internet; define cloud computing; describe the Internet of Things; describe the roles of cloud computing and cloud storage and their uses.
- **Explain the principles of security.** Describe how personal information can be used in inappropriate ways; describe how to prevent someone from gaining access to an online account; describe steps that websites take to keep passwords secure; describe the Caesar Cipher.
- **Implement encryption and authentication strategies.** Encode and decode messages using Unicode; evaluate passwords based on security criteria; describe how hashed passwords enhance Internet security.

Examine the Software Development Process

- **Plan and create programs.** Analyze problems in relation to your audience and identify which apps or games can be part of the solution and how they can be used; describe user-experience principles; gather user input; use code or text to create instructions for using a program.
- **Describe software development processes used to solve problems.** TPlan project tasks and delegate responsibilities; describe a cycle of create, evaluate, and revise.
- **Analyze and evaluate completed programs.** Evaluate for readability and usability; give and receive feedback; evaluate feedback and revise the program accordingly.