The following is a Create Task example from a lab we did in class: Pick Up Coins. It's a simple lab but satisfies all of the requirements of the Create Task. The two most important questions in this task is 2c(algorithms) and 2d(abstractions). We'll go over how to do these.

PLEASE USE THIS ONLY AS AN EXAMPLE AND DO NOT SIMPLY COPY AND PASTE. IF ANY OF YOU SIMPLY COPY AND PASTE OR ONLY MODIFY SLIGHTLY AND HAVE SIMILAR ANSWERS, YOU WILL BE REPORTED FOR PLAGIARISM.

WRITE ALL OF YOUR ANSWERS IN YOUR OWN WORDS!!

Create Task - Applications from Ideas

Written Response Template

**Program Purpose and Development**

**2a)** Provide a written response or audio narration in your video that:

1. identifies the programming language;
2. identifies the purpose of your program; and
3. explains what the video illustrates.

*(Must not exceed 150 words)*

**Note: Remember that part of your submission is to record a video. Submit one video in .mp4, .wmv, .avi, or .mov format that demonstrates the running of at least one significant feature of your program. Your video must not exceed 1 minute in length and must not exceed 30MB in size.**

**If you narrate your video, you can answer the above their questions in the video and DO NOT NEED to write the answers. However, if you don't narrate the answers, you need to write your answers in 2a).**

For example, in narration or in writing:

I use Python and the Processing IDE to write this game. The purpose of this game is for the player to collect coins on the screen. The player wins if he/she collects all coins. In the video, I illustrate the running of this game. Note that as the player collects the coins, a tally is displayed on the screen to keep track of the coins that have already been collected. The game is over once all coins are collected and a game over screen is displayed. The user can press "r" to restart the game.

**2b)** Describe the incremental and iterative development process of your program, focusing on two distinct points in that process. Describe the difficulties and/or opportunities you encountered and how they were resolved or incorporated. In your description clearly indicate whether the development described was collaborative or independent. At least one of these points must refer to independent program development. *(Must not exceed 200 words)*

**Wikipedia definition of incremental and iterative:**

The basic idea behind this method is to develop a system through repeated cycles (iterative) and in smaller portions at a time (incremental), allowing [software developers](https://en.wikipedia.org/wiki/Software_developer) to take advantage of what was learned during development of earlier parts or versions of the system. -Wikipedia

(For example, as you add more features to your game(incremental), you might find it necessary to revisit your previous code(iterative) and modify or improve them.)

**Read examples for this response from the College Board website here:**

<https://apcentral.collegeboard.org/courses/ap-computer-science-principles/exam>

**2c)** Capture and paste a program code segment that implements an algorithm (marked with an **oval** in **section 3** below) and that is fundamental for your program to achieve its intended purpose. This code segment must be an algorithm you developed individually on your own, must include two or more algorithms, and must integrate mathematical and/or logical concepts. Describe how each algorithm within your selected algorithm functions independently, as well as in combination with others, to form a new algorithm that helps to achieve the intended purpose of the program. *(Must not exceed 200 words)*

A screenshot of a cell phone

Description automatically generated

My main algorithm is located in the on\_update method where the game screen and all objects are updated sixty times a second. I use a logical condition which checks that the game is not currently over. If so, the game proceeds as defined by three subalgorithms: 1) updating the player, 2) checking for coin collisions and 3) checking for game over.

I will now explain how these algorithms work independently and how together they help me achieve the intended purpose of my game. The update method called on the player object is part of the Python Arcade library which simply adds the velocity of the player to its position to move it. I wrote the check\_coin\_collision method which checks for any collision between the player and the list of coins in self.coin\_list. If there exists at least one collision, loop through each coin in the collision list and add 1 to the tally self.num\_coins variable and remove the coin from self.coin\_list. After collisions are resolved, the check\_game\_over method is called to check if all of the coins have been collected by the player and update the self.game\_over state. These subalgorithms allow the player object to collect coins as intended.

**2d)** Capture and paste a program code segment that contains an abstraction you developed individually on your own (marked with a **rectangle** in **section 3** below). This abstraction must integrate mathematical and logical concepts. **Explain how your abstraction helped manage the complexity of your program.** *(Must not exceed 200 words)*

A screenshot of a cell phone

Description automatically generated



A screenshot of a cell phone

Description automatically generated

My abstraction is a list of coin Sprite objects found in the variable self.coin\_list. This abstraction allows me to effectively keep track of all of the coins in the game in one variable and manage the complexity of my game. I initialize this list by iterating through a loop and creating a coin Sprite object as defined by the Python Arcade library. The coin Sprite is then placed randomly on the screen by calling the randrange method of the random library and appended to my self.coin\_list.

This one variable allows me to store potentially hundreds of coin objects and simplifies, for example, the process of drawing them by calling one command, self.coin\_list.draw(), to draw all of them simultaneously. In addition, it allows me to easily compute the collision list between the player and the list of coins at each frame. If this collision list is not empty, I can loop through it and update the score by adding 1 for each coin in the collision\_list and remove it from self.coin\_list. In the next frame, when self.coin\_list.draw() is called, coins that have been removed from the list will not be drawn and my game is properly updated.

REMEMBER to also generate a PDF of your full code with oval and rectangle highlighting the appropriate part of the code. This is a separate submission.