**Create – Applications From Ideas  
Written Response Submission Template**

Please see [Assessment Overview and Performance Task Directions for Student](https://apcentral.collegeboard.org/pdf/ap-csp-student-task-directions.pdf?course=ap-computer-science-principles) for the task directions and recommended word counts.

**Program Purpose and Development**

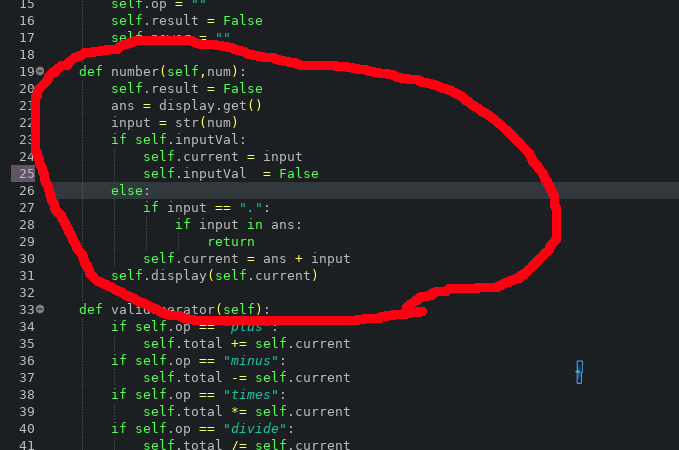
2a)

|  |
| --- |
| My program was developed using the Python programming language. The purpose of the program is to provide a fast and simple calculator with advanced functions like square root and cube root as well as constants like pi and e. The video is meant to illustrate using a variety of functions in the calculator and demonstrate how the calculator can preserve the expression while the display clears when you press a function button. |

2b)

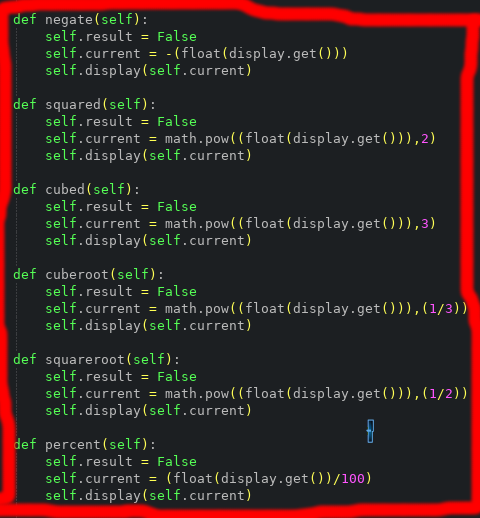
|  |
| --- |
| An iterative process I used in my program can be seen in the negate function for my calculator. First it sets the result boolean to false, which is useful for the functionality the operator function. After this, it grabs the float of what is in the display, negates it, and puts it int the “current” variable. The current variable then has its value passed to the display again. This methodology is useful for a variety of functions on the calculator such as the roots and powers, which instead of negating, use the math.pow function provided by python’s math plugin. In my number method, I first save the display value in ans, and then check if inputVal is true. If it is, that means a new number needs to be added to the display. If not, I can keep adding to the number by adding the input to the ans variable from before. One difficulty I encountered was when I was making my power functions. I tried combining them into a single function with a pow variable as an input from the button, but I experienced issued with that and converted them into separate functions. I tested the calculator myself. |

2c)



|  |
| --- |
| An algorithm in my program is my number function which includes some preceding declarations, an if and else statement, and a display statement at the end. The point of this particular function is to provide functionality for the number buttons in the calculator. The first if statement checks if the calculator has just taken an operator and needs to input a new value. If so, it will set the current to the input and set inputVal back to false. The else statement will be used in most other circumstances where a number will already be in the calculator and it will just need to combine the current and input using the mathematical concept of addition and add the input to the string. With all of this, the number algorithm lets the calculator take number inputs from the user in a variety of situations. This algorithm was developed independently. |

2d)



|  |
| --- |
| An abstraction that I developed individually was the negate function, percent function, and the power functions. These functions integrate mathematical functions by multiplying, dividing, or applying a power to the float of the current display. These help reduce complexity by allowing the user to have access to normally difficult math functions as the click of the button. Separating them into different functions also makes the code easier to debug because each function is smaller and easier to look at and find mistakes. The separated functions also makes it so that I can name them for exactly what they do, which is helpful when looking at what the code does without too much reading. |