

## Create – Applications From Ideas

### Written Response Submission Template

Please see [Assessment Overview and Performance Task Directions for Student](#) for the task directions and recommended word counts.

#### Program Purpose and Development

2a)

The program I have created, called "Simple Command Line Calculator", is made using the Microsoft Small Basic (MSB) programming language. The purpose of the program is to allow users to summon and use a simple calculator that incorporates all of the four basic mathematical functions: addition, subtraction, multiplication, and division. It does so through a simple interface which makes use of only the numbered keys on the user's keyboard, for simplicity. The video companion to this project demonstrates the use of each of the four functions, as well as the fail safe which prevents the user from dividing by zero, and the end sequence.

2b)

The incremental and iterative process of this program's development was rather simple. The entire program was coded by myself, using the Microsoft Small Basic reference materials which are available on the official MSDN website. For each iteration, I implemented one of the main functions of the program. For example, my first step was implementing an operation which would allow me to return to the intro portion of the program at any time without having to needlessly copy and paste large amounts of code. And, at the end of the program, I focused on implementing the division part of the calculator portion of the program. This last part was more difficult than any of the other parts because I quickly realized that the program would overload and crash upon the user attempting to divide by zero. I ultimately decided to overcome this issue by implementing a fail safe which prevents the user from even attempting to divide by zero, thus eliminating the chance that the program will attempt said division. At the end of the program, there were about ten different iterations.

2c)

```

Sub division
    TextWindow.Clear()
    TextWindow.WriteLine("You have chosen division.")
    TextWindow.WriteLine("Please enter your first number below.")
    TextWindow.WriteLine("")
    TextWindow.WriteLine("")
    num1div = TextWindow.ReadNumber()
    checkforzero()
EndSub

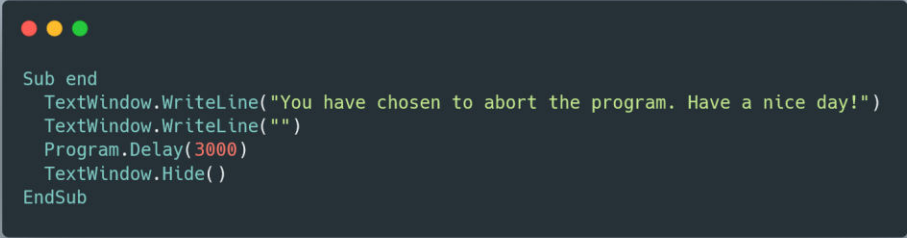
Sub checkforzero
    TextWindow.Clear()
    TextWindow.WriteLine("Your first number is " + num1div + ". Please enter a second number.")
    TextWindow.WriteLine("")
    TextWindow.WriteLine("")
    num2div = TextWindow.ReadNumber()
    If num2div = 0 Then
        TextWindow.Clear()
        TextWindow.WriteLine("Error: You cannot attempt to divide by 0. Please wait to try again.")
        TextWindow.WriteLine("")
        Program.Delay(1500)
        TextWindow.Clear()
        checkforzero()
    ElseIf num2div <> 0 then
        dividewithoutzero()
    EndIf
EndSub

Sub dividewithoutzero
    TextWindow.Clear()
    num3div = num1div / num2div
    TextWindow.WriteLine("Your new number is " + num3div + ".")
    TextWindow.WriteLine("")
    TextWindow.WriteLine("Would you like to start over? y/n")
    TextWindow.WriteLine("")
    addsel = TextWindow.Read()
    If addsel = "y" Then
        TextWindow.Clear()
        intro()
    EndIf
    If addsel = "n" Then
        TextWindow.Clear()
        end()
    EndIf
EndSub

```

The algorithm I have chosen is the calculator's division algorithm. It is split up into multiple operations which are called depending on the progress of the activity, but when put together it contains multiple algorithms within it that serve to make the algorithm as a whole function without error. The first self-contained algorithm is an if/then/else statement (which is called by the operator "checkforzero()") that checks to see if the second number that the user has entered is equal to zero. If it is, in fact, equal to zero, the program will disallow the division and instead prompt the user to enter a different number that isn't zero. If the number is not zero, it allows the division to take place as usual (through the calling of another function which contains the rest of the division algorithm). Another algorithm within this main algorithm is the function that asks the user whether or not they want to go back to the beginning of the program to complete more functions. If yes, it calls the intro function and sends the user to the start. If no, it calls the end function and closes the program window.

2d)



```
Sub end
    TextWindow.WriteLine("You have chosen to abort the program. Have a nice day!")
    TextWindow.WriteLine("")
    Program.Delay(3000)
    TextWindow.Hide()
EndSub
```

The abstraction I have chosen is the "end" operation. It takes the form of a small algorithm that handles the closing of the program window with only four simple lines of code. This simple abstraction allowed me to call a universal program end sequence from anywhere else within the code, which saved me from having to copy and paste the same code in multiple places where the user might be prompted to choose whether or not to end the program. It integrates logical and mathematical concepts through its simple wait time operation, which uses a set number of milliseconds (3000 in this case) to know how long to wait after displaying the messages in the first two lines before the program window closes.