CS213: Programming II

Assignment 1

Cairo University, Faculty of Computers
and Artificial Intelligence

FACULTY OF COMPUTERS AND ARTIFICIAL INTELLIGENCE - CAIRO UNIVERSITY

CS213: Programming II

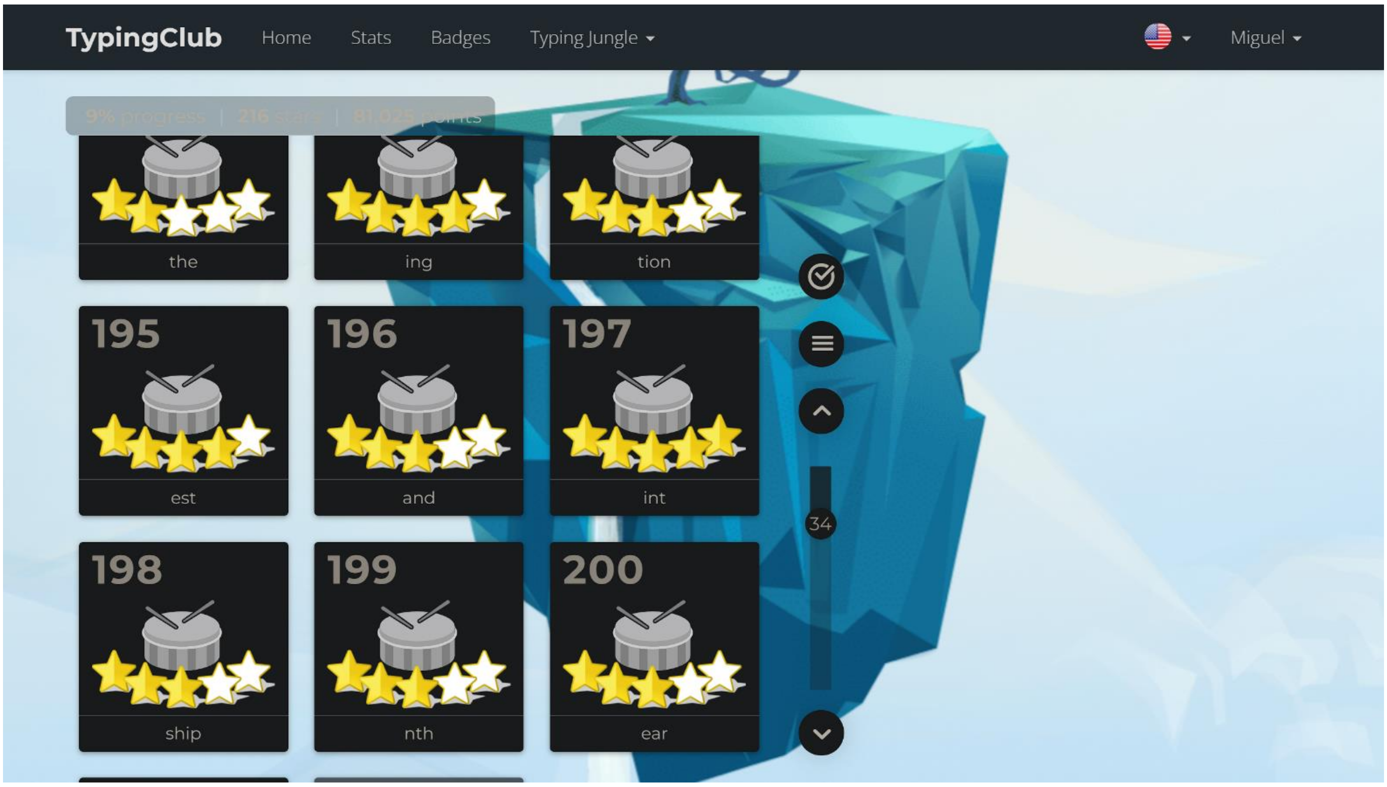
Assignment 1

Course Professor:

Dr. Mohammed El-Ramly

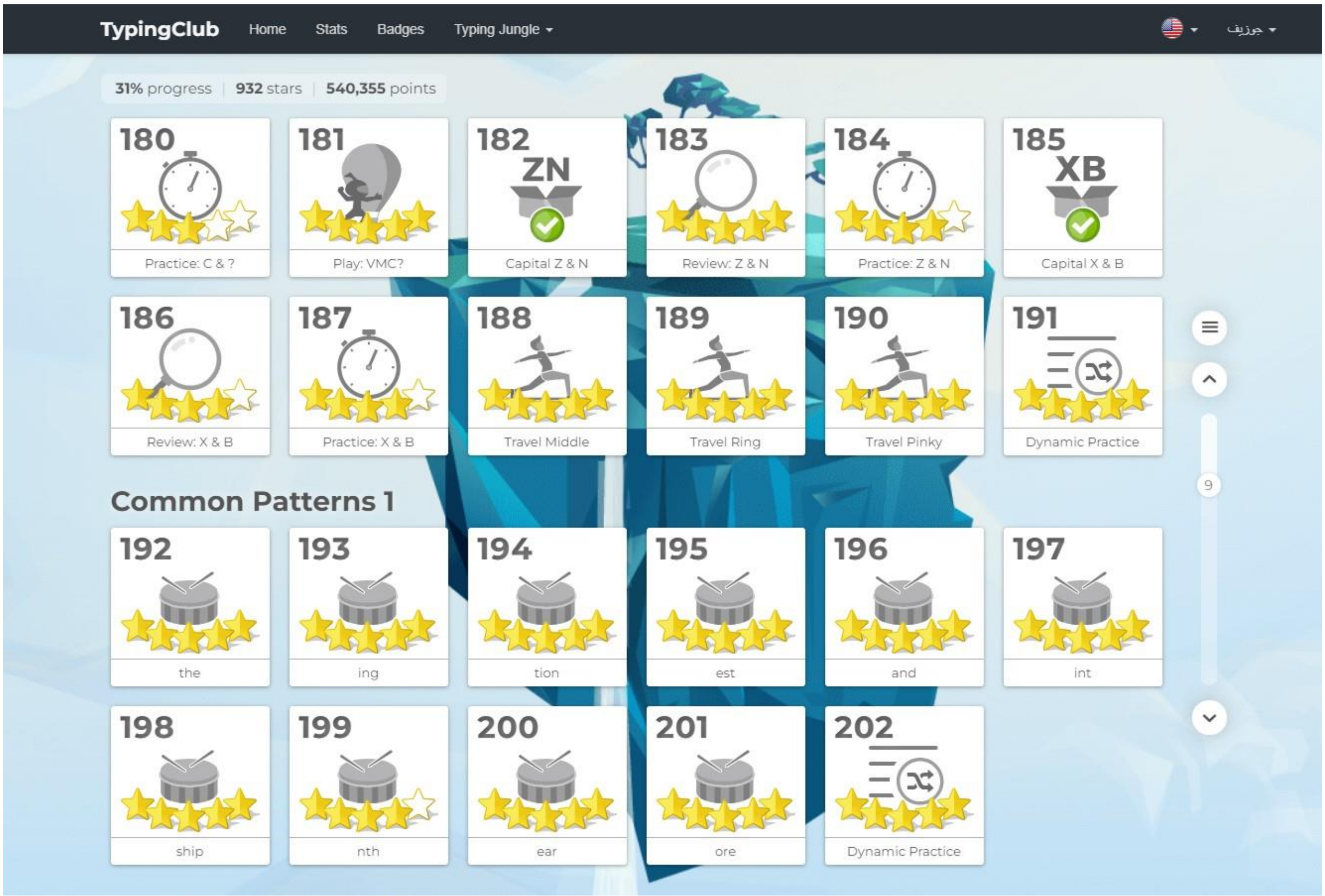
Prepared by:

• Joseph Sameh	S10	20220099	jojo.1922005@gmail.com
• Miguel Reda	S9	20220352	megooreda2005@gmail.com
• Youssef Joseph	S19	20220389	youssefjoseph35@gmail.com



9% progress | 216 stars | 81,025 points

the
ing
tion
195 est
196 and
197 int
198 ship
199 nth
200 ear

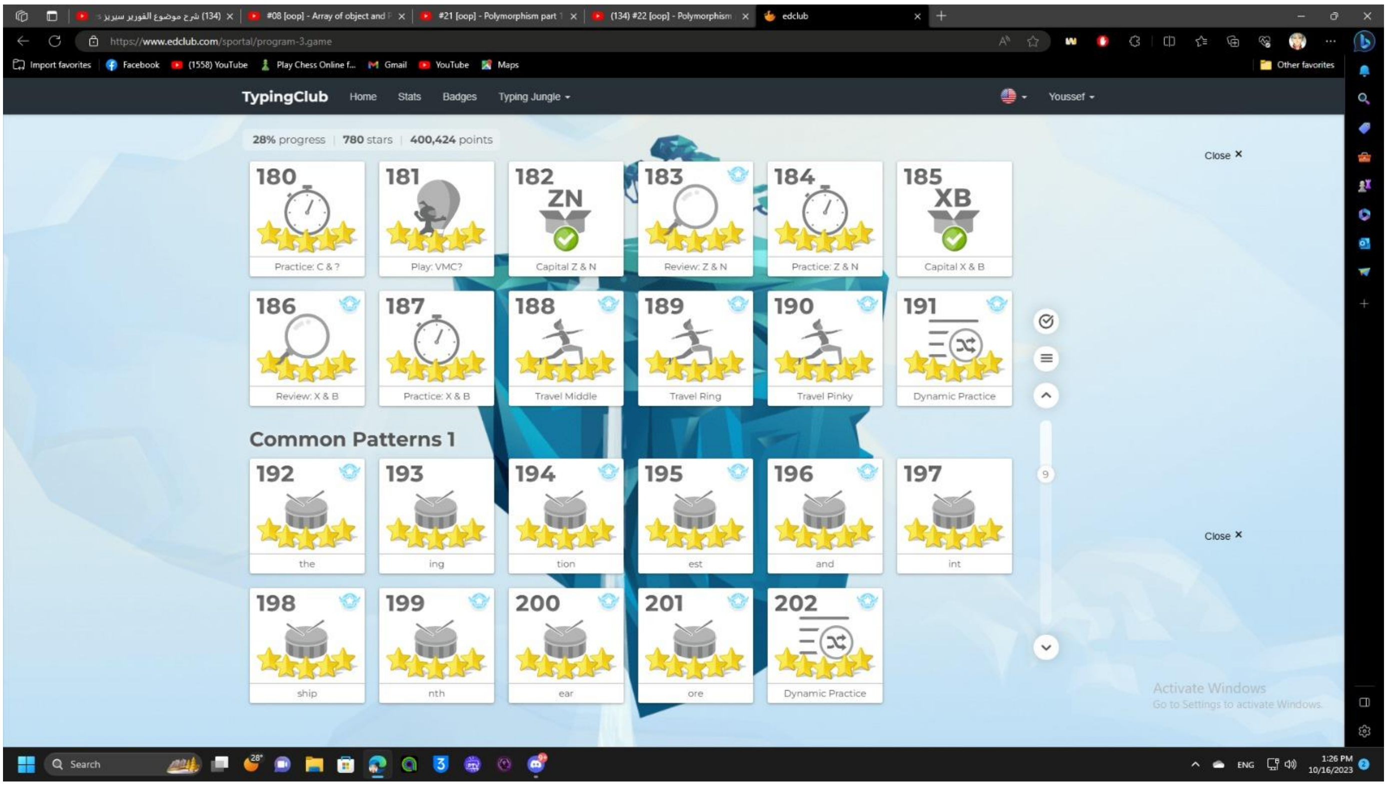


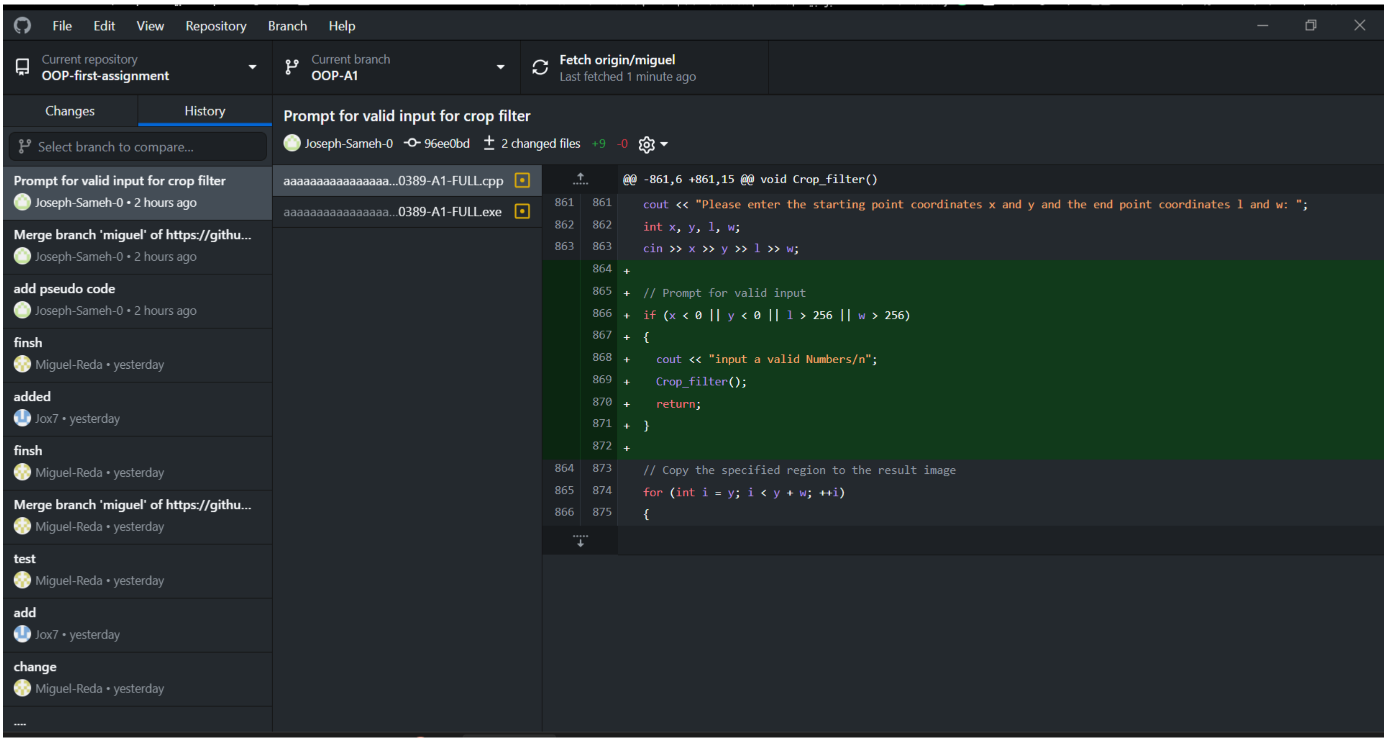
31% progress | 932 stars | 540,355 points

180 Practice: C & ?
181 Play: VMC?
182 ZN Capital Z & N
183 Review: Z & N
184 Practice: Z & N
185 XB Capital X & B
186 Review: X & B
187 Practice: X & B
188 Travel Middle
189 Travel Ring
190 Travel Pinky
191 Dynamic Practice

Common Patterns 1

192 the
193 ing
194 tion
195 est
196 and
197 int
198 ship
199 nth
200 ear
201 ore
202 Dynamic Practice





Prompt for valid input for crop filter
Joseph-Sameh-0 • 2 hours ago

Merge branch 'miguel' of https://githu...
Joseph-Sameh-0 • 2 hours ago

add pseudo code
Joseph-Sameh-0 • 2 hours ago

finsh
Miguel-Reda • yesterday

added
Jox7 • yesterday

finsh
Miguel-Reda • yesterday

Merge branch 'miguel' of https://githu...
Miguel-Reda • yesterday

test
Miguel-Reda • yesterday

add
Jox7 • yesterday

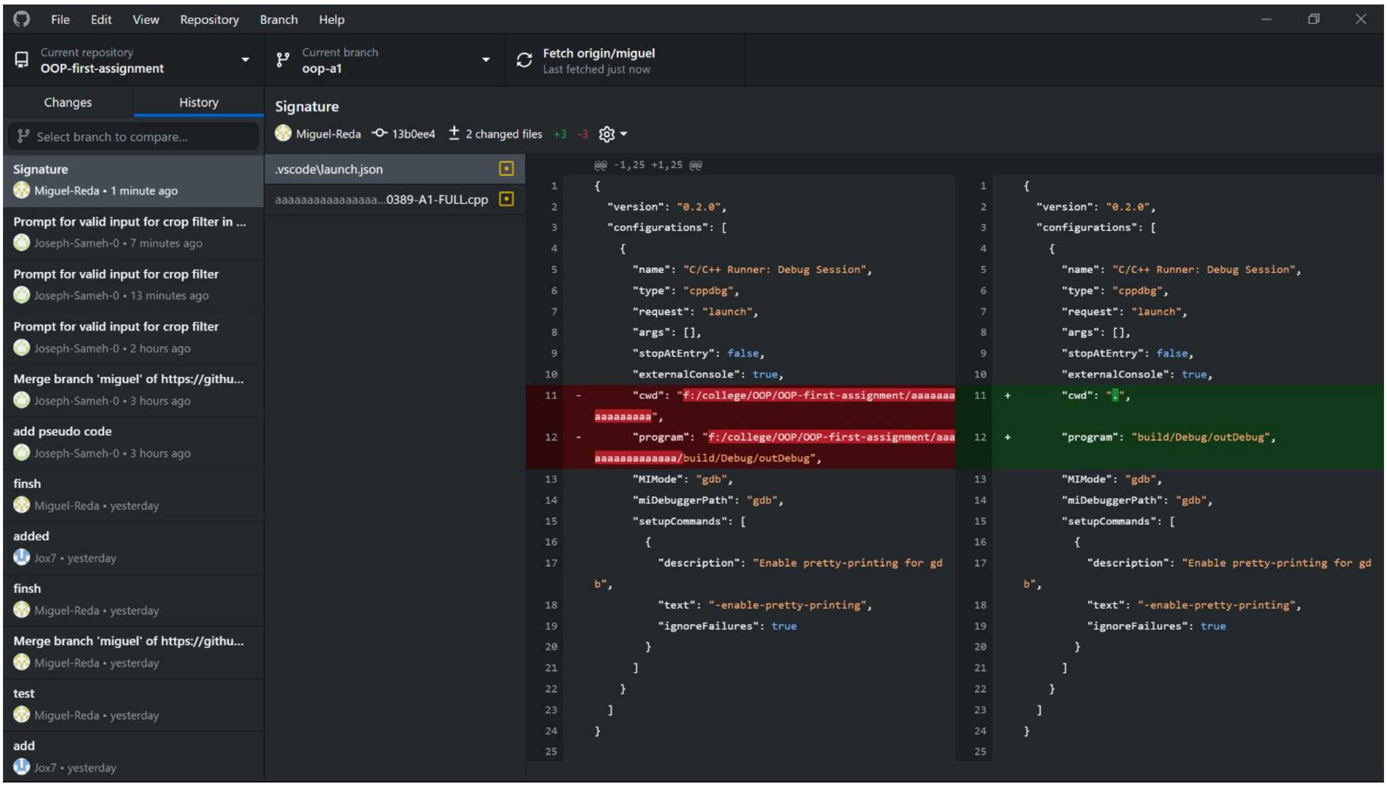
change
Miguel-Reda • yesterday

Prompt for valid input for crop filter

Joseph-Sameh-0 96ee0bd 2 changed files +9 -0

aaaaaaaaaaaaaaaa...0389-A1-FULL.cpp
aaaaaaaaaaaaaaaa...0389-A1-FULL.exe

```
@@ -861,6 +861,15 @@ void Crop_filter()
    cout << "Please enter the starting point coordinates x and y and the end point coordinates l and w: ";
    int x, y, l, w;
    cin >> x >> y >> l >> w;
+
+   // Prompt for valid input
+   if (x < 0 || y < 0 || l > 256 || w > 256)
+   {
+     cout << "input a valid Numbers/n";
+     Crop_filter();
+     return;
+   }
+
    // Copy the specified region to the result image
    for (int i = y; i < y + w; ++i)
    {
```



Signature
Miguel-Reda • 1 minute ago

Prompt for valid input for crop filter in ...
Joseph-Sameh-0 • 7 minutes ago

Prompt for valid input for crop filter
Joseph-Sameh-0 • 13 minutes ago

Prompt for valid input for crop filter
Joseph-Sameh-0 • 2 hours ago

Merge branch 'miguel' of https://githu...
Joseph-Sameh-0 • 3 hours ago

add pseudo code
Joseph-Sameh-0 • 3 hours ago

finsh
Miguel-Reda • yesterday

added
Jox7 • yesterday

finsh
Miguel-Reda • yesterday

Merge branch 'miguel' of https://githu...
Miguel-Reda • yesterday

test
Miguel-Reda • yesterday

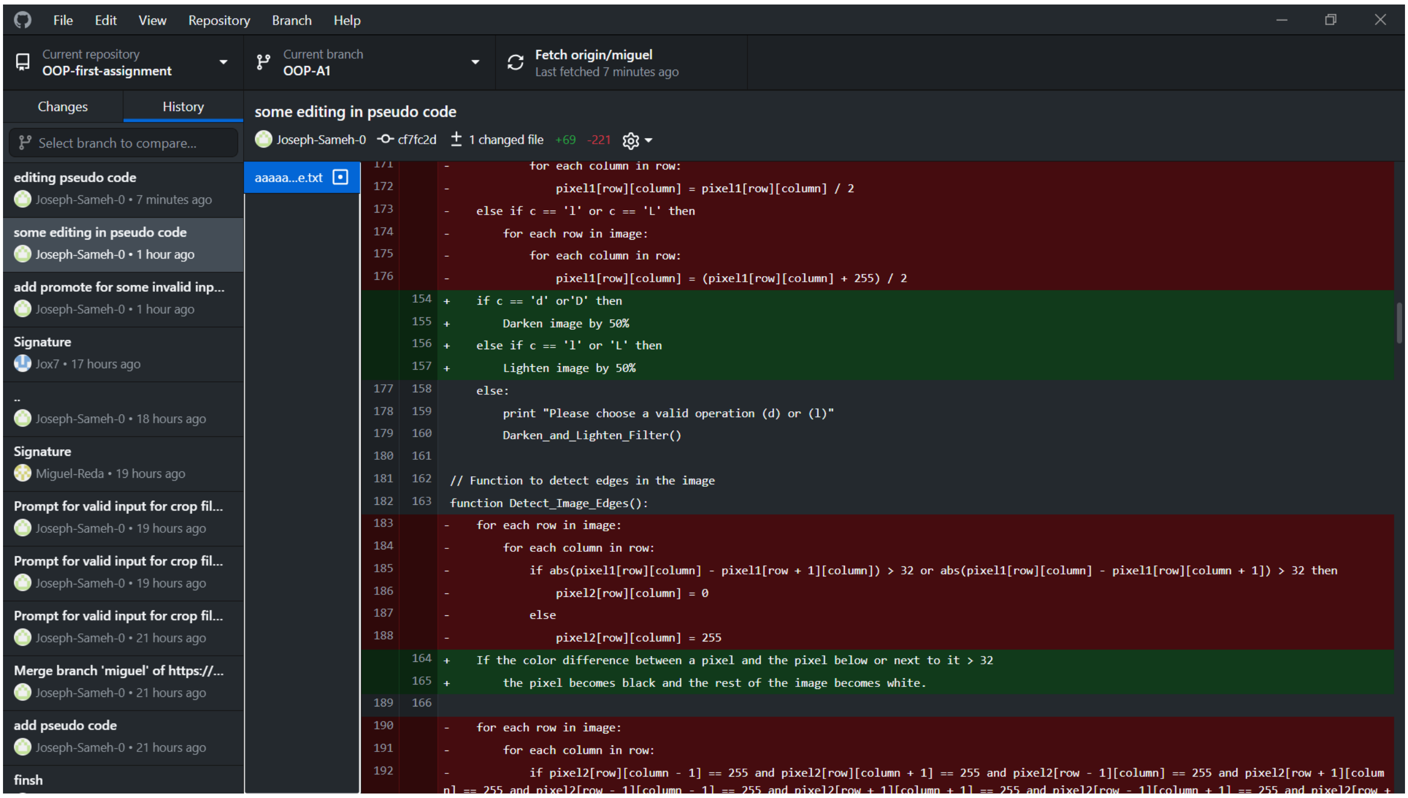
add
Jox7 • yesterday

Signature
Miguel-Reda 13b0ee4 2 changed files +3 -3

.vscode\launch.json
aaaaaaaaaaaaaaaa...0389-A1-FULL.cpp

```
@@ -1,25 +1,25 @@
{
  "version": "0.2.0",
  "configurations": [
    {
      "name": "C/C++ Runner: Debug Session",
      "type": "cppdbg",
      "request": "launch",
      "args": [],
      "stopAtEntry": false,
      "externalConsole": true,
-     "cwd": "f:/college/OOP/OOP-first-assignment/aaaaaaaaaaaaaaaa",
-     "program": "f:/college/OOP/OOP-first-assignment/aaaaaaaaaaaaaaaa/build/Debug/outDebug",
      "MIMode": "gdb",
      "miDebuggerPath": "gdb",
      "setupCommands": [
        {
          "description": "Enable pretty-printing for gdb",
          "text": "-enable-pretty-printing",
          "ignoreFailures": true
        }
      ]
    }
  ]
}
```

```
{
  "version": "0.2.0",
  "configurations": [
    {
      "name": "C/C++ Runner: Debug Session",
      "type": "cppdbg",
      "request": "launch",
      "args": [],
      "stopAtEntry": false,
      "externalConsole": true,
+     "cwd": ".",
+     "program": "build/Debug/outDebug",
      "MIMode": "gdb",
      "miDebuggerPath": "gdb",
      "setupCommands": [
        {
          "description": "Enable pretty-printing for gdb",
          "text": "-enable-pretty-printing",
          "ignoreFailures": true
        }
      ]
    }
  ]
}
```



editing pseudo code
Joseph-Sameh-0 • 7 minutes ago

some editing in pseudo code
Joseph-Sameh-0 • 1 hour ago

add promote for some invalid inp...
Joseph-Sameh-0 • 1 hour ago

Signature
Jox7 • 17 hours ago

..
Joseph-Sameh-0 • 18 hours ago

Signature
Miguel-Reda • 19 hours ago

Prompt for valid input for crop fil...
Joseph-Sameh-0 • 19 hours ago

Prompt for valid input for crop fil...
Joseph-Sameh-0 • 19 hours ago

Prompt for valid input for crop fil...
Joseph-Sameh-0 • 21 hours ago

Merge branch 'miguel' of https://...
Joseph-Sameh-0 • 21 hours ago

add pseudo code
Joseph-Sameh-0 • 21 hours ago

finsh

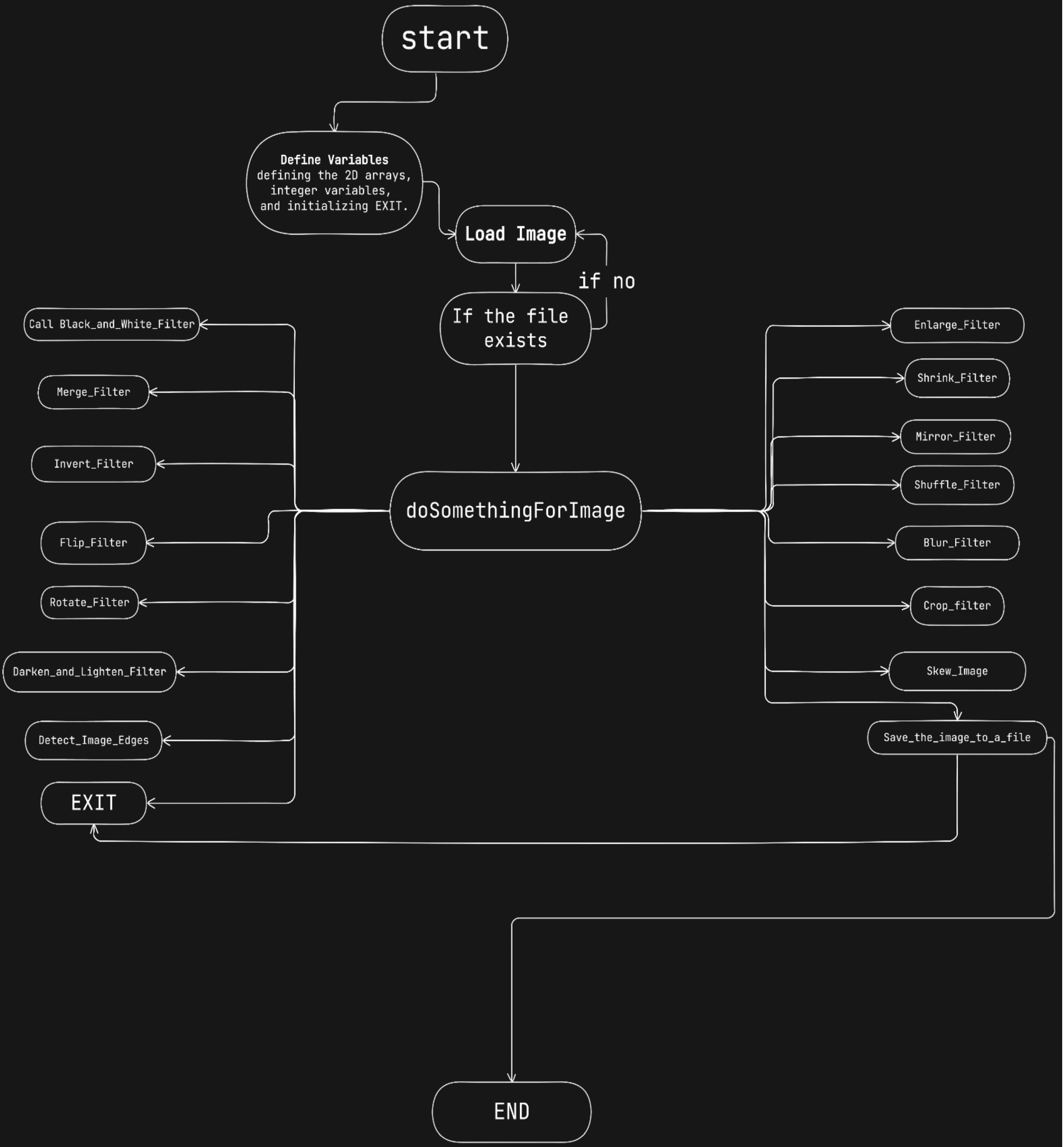
some editing in pseudo code
Joseph-Sameh-0 cf7fc2d 1 changed file +69 -221

aaaaa...e.txt

```
-           for each column in row:
-               pixel1[row][column] = pixel1[row][column] / 2
-   else if c == 'l' or c == 'L' then
-       for each row in image:
-           for each column in row:
-               pixel1[row][column] = (pixel1[row][column] + 255) / 2
+   if c == 'd' or'D' then
+       Darken image by 50%
+   else if c == 'l' or 'L' then
+       Lighten image by 50%
    else:
        print "Please choose a valid operation (d) or (l)"
        Darken_and_Lighten_Filter()

// Function to detect edges in the image
function Detect_Image_Edges():
-   for each row in image:
-       for each column in row:
-           if abs(pixel1[row][column] - pixel1[row + 1][column]) > 32 or abs(pixel1[row][column] - pixel1[row][column + 1]) > 32 then
-               pixel2[row][column] = 0
-           else
-               pixel2[row][column] = 255
+   If the color difference between a pixel and the pixel below or next to it > 32
+       the pixel becomes black and the rest of the image becomes white.

-   for each row in image:
-       for each column in row:
-           if pixel2[row][column - 1] == 255 and pixel2[row][column + 1] == 255 and pixel2[row - 1][column] == 255 and pixel2[row + 1][column] == 255 and pixel2[row - 1][column - 1] == 255 and pixel2[row + 1][column + 1] == 255 and pixel2[row - 1][column + 1] == 255 and pixel2[row +
```


start
Define Variables
defining the 2D arrays,
integer variables,
and initializing EXIT.
Load Image
if no
If the file exists
doSomethingForImage
Call Black_and_White_Filter
Merge_Filter
Invert_Filter
Flip_Filter
Rotate_Filter
Darken_and_Lighten_Filter
Detect_Image_Edges
EXIT
Enlarge_Filter
Shrink_Filter
Mirror_Filter
Shuffle_Filter
Blur_Filter
Crop_filter
Skew_Image
Save_the_image_to_a_file
END

```
Define 2D array image1 of unsigned char with 256 x 256 dimensions // First image
Define 2D array image2 of unsigned char with 256 x 256 dimensions // Second image
Define integer variable EXIT and set it to 1 // Program exit control

Define function loadImage taking an argument image, a 2D array of unsigned char
    Read user input into imageFileName // Read image file name from user
    Concatenate ".bmp" to imageFileName
    Read grayscale image from imageFileName into image
    If imageFileName does not exist in the file system
        Prompt user to enter a valid file name of the image
        Call loadImage with image as argument

Define function saveImage taking an argument image, a 2D array of unsigned char
    Read user input into imageFileName // Read target image file name from user
    Concatenate ".bmp" to imageFileName
    Write image to imageFileName

Define filter functions for image processing:
    Black_and_White_Filter()
    Invert_Filter()
    Merge_Filter()
    Flip_Filter()
    Rotate_Filter()
    Darken_and_Lighten_Filter()
    Detect_Image_Edges()
    Enlarge_Filter()
    Shrink_Filter()
    Mirror_Filter()
    Shuffle_Filter()
    Blur_Filter()
    Crop_filter()
    Skew_Image_Right()
    Skew_Image_Up()
    Save_the_image_to_a_file()
    copy_image_to_image1()
    Exit()

Define function doSomethingForImage()
    Display menu options
    Read user's choice into choose_op
    Switch choose_op
        Case '1': Call Black_and_White_Filter()
        Case '2': Call Invert_Filter()
        Case '3': Merge_Filter();
        Case '4': Flip_Filter();
        Case '5': Rotate_Filter();
        Case '6': Darken_and_Lighten_Filter();
        Case '7': Detect_Image_Edges();
        Case '8': Enlarge_Filter();
        Case '9': Shrink_Filter();
        Case 'a': Mirror_Filter();
        Case 'b': Shuffle_Filter();
        Case 'c': Blur_Filter();
        Case 'd': Crop_filter();
        Case 'e': Skew_Image_Right();
        Case 'f': Skew_Image_Up();
        Case 's': Call Save_the_image_to_a_file()
        Case '0': Call Exit()
        Default: Print "please, select a valid operation" and Call doSomethingForImage()
```

```
Define function main()
    Print "Please enter file name of the image to process: "
    Call loadImage with image1 as argument
    Call doSomethingForImage()
    While EXIT is true
        Print "Do you want to do another operation on the image? (y) (n)"
        Read choice
        If choice is 'y' or 'Y'
            Call doSomethingForImage()
        Else if choice is 'n' or 'N'
            Print "s- Save the image to a file"
            Print "0- Exit"
            Print "else- back"
            Read choice
            Switch choice
                Case 's': Call Save_the_image_to_a_file()
                Case '0': Set EXIT to 0
        Else
            Print "please, select a valid operation"

Define function loadImage taking an argument image, a 2D array of unsigned char
    Read imageFileName // Read gray scale image file name from user
    Concatenate ".bmp" to imageFileName
    Read grayscale image from imageFileName into image
    If imageFileName does not exist in the file system
        Print "Please enter a valid file name of the image to process: "
        Call loadImage with image as argument

Define function saveImage taking an argument image, a 2D array of unsigned char
    Read imageFileName // Read target image file name from user
    Concatenate ".bmp" to imageFileName
    Write image to imageFileName



// Function for Black & White Filter
function Black_and_White_Filter():
    for each row in image:
        for each pixel in row:
            if pixel > 127 then
                pixel = 255
            else
                pixel = 0

// Function for Invert Filter
function Invert_Filter():
    for each row in image:
        for each pixel in row:
            pixel = 255 - pixel

// Function for Merge Filter
function Merge_Filter():
    print "Enter the 2nd source image file name: "
    loadImage(image2) // Load the second image

    for each row in image:
        for each pixel in row:
            pixel = (pixel1 + pixel2) / 2
```

```
// Function for Flip Filter
function Flip_Filter():
    print "Flip (h)orizontally or (v)ertically ?: "
    input char character
    if character == 'h' or 'H' then
        Flip the image horizontally

    else if character == 'v' or 'V' then
        Flip the image vertically
    else:
        print "please, select a valid operation (v) or (h)"
        Flip_Filter()

// Function for Rotate Filter
function Rotate_Filter():
    print "Rotate (90), (180) or (270) degrees?: "
    int b = input()
    b = b % 360 // Ensure b is within 0-359 range
    if b == 0 then
        // No rotation required
    else if b == 90 then
        Rotate image by 90 degrees
    else if b == 180 then
        Rotate image by 180 degrees
    else if b == 270 then
        Rotate image by 270 degrees
    else:
        print "Please choose a number that is divisible by 90"
        Rotate_Filter()

// Function for Darken & Lighten Filter
function Darken_and_Lighten_Filter():
    print "Do you want to (d)arken or (l)ighten ?: "
    char c = input()
    if c == 'd' or'D' then
        Darken image by 50%
    else if c == 'l' or 'L' then
        Lighten image by 50%
    else:
        print "Please choose a valid operation (d) or (l)"
        Darken_and_Lighten_Filter()

// Function to detect edges in the image
function Detect_Image_Edges():
    If the color difference between a pixel and the pixel below or next to it > 32
        the pixel becomes black and the rest of the image becomes white.

    Delete single points
```

```
// Function to enlarge a quarter of the image
function Enlarge_Filter():
    print "Which quarter to enlarge 1, 2, 3 or 4?"
    input int choose
    if choose == 1 then
        Extract the top-left quarter and enlarge it
    else if choose == 2 then
        Extract the top-right quarter and enlarge it
    else if choose == 3 then
        Extract the down-left quarter and enlarge it
    else if choose == 4 then
        Extract the down-right quarter and enlarge it
    else:
        print "Invalid choice, please choose 1, 2, 3, or 4"
        Enlarge_Filter()

// Function to shrink the image
FUNCTION Shrink_Filter()
    PRINT "Shrink to (1/2), (1/3) or (1/4)?"
    INPUT shrink

    IF shrink EQUALS "1/2" THEN
        Shrink the image by half
    ELSE IF shrink EQUALS "1/3" THEN
        Shrink the image by one-third
    ELSE IF shrink EQUALS "1/4" THEN
        Shrink the image by one-fourth
    ELSE
        PRINT "input a valid value (1/2), (1/3) or (1/4)"
        CALL Shrink_Filter() // Prompt for valid input


// Function to mirror the image
FUNCTION Mirror_Filter()
    PRINT "Mirror (l)eft, (r)ight, (u)pper, (d)own side?"
    INPUT mirror

    IF mirror EQUALS 'l' OR 'L' THEN
        Mirror the image horizontally (left to right)
    ELSE IF mirror EQUALS 'r' OR 'R' THEN
        Mirror the image horizontally (right to left)
    ELSE IF mirror EQUALS 'u' OR 'U' THEN
        Mirror the image vertically (upper to lower)
    ELSE IF mirror EQUALS 'd' OR 'D' THEN
        Mirror the image vertically (lower to upper)
    ELSE
        PRINT "input a valid character l, r, u or d"
        CALL Mirror_Filter() // Prompt for valid input
```

```
// Function to shuffle the quarters of the image
FUNCTION Shuffle_Filter()
    PRINT "New order of quarters ?: "
    ARRAY order[4]

    FOR each index in order
        INPUT order

        IF order LESS THAN 1 OR order GREATER THAN 4 THEN
            PRINT "Invalid input. Please enter numbers between 1 and 4."
            Clear input buffer
            CALL Shuffle_Filter()
            RETURN
    ENDFOR

    IF order is {1, 2, 3, 4} THEN
        RETURN
    ELSE
        INTEGER current quarter = 1

        FOR EACH index IN order
            IF index EQUALS 1 THEN
                IF current quarter EQUALS 1 THEN
                    Copy the top-left quarter to the top-left quarter in result image
                ELSE IF current quarter EQUALS 2 THEN
                    Copy the top-right quarter to the top-left quarter in result image
                ELSE IF current quarter EQUALS 3 THEN
                    Copy the down-left quarter to the top-left quarter in result image
                ELSE IF current quarter EQUALS 4 THEN
                    Copy the down-right quarter to the top-left quarter in result image
                ENDIF
            ELSE IF index EQUALS 2 THEN
                IF current quarter EQUALS 1 THEN
                    Copy the top-left quarter to the top-right quarter in result image
                ELSE IF current quarter EQUALS 2 THEN
                    Copy the top-right quarter to the top-right quarter in result image
                ELSE IF current quarter EQUALS 3 THEN
                    Copy the down-left quarter to the top-right quarter in result image
                ELSE IF current quarter EQUALS 4 THEN
                    Copy the down-right quarter to the top-right quarter in result image
                ENDIF
            ELSE IF index EQUALS 3 THEN
                IF current quarter EQUALS 1 THEN
                    Copy the top-left quarter to the down-left quarter in result image
                ELSE IF current quarter EQUALS 2 THEN
                    Copy the top-right quarter to the down-left quarter in result image
                ELSE IF current quarter EQUALS 3 THEN
                    Copy the down-left quarter to the down-left quarter in result image
                ELSE IF current quarter EQUALS 4 THEN
                    Copy the down-right quarter to the down-left quarter in result image
                ENDIF
            ELSE IF index EQUALS 4 THEN
                IF current quarter EQUALS 1 THEN
                    Copy the top-left quarter to the down-right quarter in result image
                ELSE IF current quarter EQUALS 2 THEN
                    Copy the top-right quarter to the down-right quarter in result image
                ELSE IF current quarter EQUALS 3 THEN
                    Copy the down-left quarter to the down-right quarter in result image
                ELSE IF current quarter EQUALS 4 THEN
                    Copy the down-right quarter to the down-right quarter in result image
                ENDIF
            ENDIF

            current quarter = current quarter + 1
        ENDFOR
    ENDIF

// Function to apply a blur filter to the image
FUNCTION Blur_Filter()
    FOR each row in image
        FOR each pixel in row
            Merge the pixel with the 8 pixels around it
    Repeat the process for better results

// Function to crop a region of interest from the image
FUNCTION Crop_Filter()
    PRINT "Please enter the starting point coordinates x and y and the end point coordinates l and w: "
    INPUT x, y, l, w

    FOR row = y TO y + w
        FOR column = x TO x + l
            Copy the specified region to the result image

// Function to skew the image to the right
FUNCTION Skew_Image_Right()
    PRINT "Please enter degree to skew right less than 89: "
    INPUT degree
    mov = tan((degree * 22) / (180 * 7)) * 256
    step = mov / 256 // Number of steps
    Initialize a temporary image (temp) with size = [256][256 + mov]

    FOR each row in image
        FOR each pixel in row + mov
            Initialize the temporary image with white pixels

    FOR each row in image
        FOR each pixel in row
            Shift the pixels to the right according to the degree of skew
        ENDFOR
        mov = mov - step
    ENDFOR
```

```
// Function to skew the image upwards
FUNCTION Skew_Image_Up()
    PRINT "Please enter degree to skew up less than 89: "
    INPUT degree
    mov = tan((degree * 22) / (180 * 7)) * 256
    step = mov / 256 // Number of steps
    Initialize a temporary image (temp) with size = [256][256 + mov]

    FOR each row in image
        FOR each pixel in row + mov
            Initialize the temporary image with white pixels


    FOR each row in image
        FOR each pixel in row
            Shift the pixels upwards according to the degree of skew
        ENDFOR
        mov = mov - step
    ENDFOR

    Copy the result back to the original image

FUNCTION to Save the image to a file()
    CALL saveImage(image1) // Save the image
    EXIT = 0 // Set EXIT flag to exit the program

FUNCTION to copy image to image1(ARRAY image[256][256])
    FOR each row in image
        FOR each pixel in row
            image1[row][column] = image[row][column]


FUNCTION TO Exit()
    PRINT "Do you want to save the image before exiting? (y) or (n)"
    INPUT Character

    IF Character EQUALS 'Y' OR 'y' THEN
        CALL saveImage(image1)
        EXIT = 0 // Set EXIT flag to exit the program
    ELSE IF Character EQUALS 'N' OR 'n' THEN
        EXIT = 0 // Set EXIT flag to exit the program
    ELSE
        PRINT "input a valid Character (y) or (n)"
        CALL Exit() // Prompt again for valid input
    ENDIF
```