

```

def foo():
    return 0



---


1 string title = "This is a Unicode pi in the sky"
2 /*
3  Defined as  $\pi = \lim_{n \rightarrow \infty} \frac{P_n}{d}$  where  $P$  is the perimeter
4  of an  $n$ -sided regular polygon circumscribing a
5  circle of diameter  $d$ .
6  */
7 const double pi = 3.1415926535



---


#  $\sum_{i=1}^{\infty} i + 1$ 
def foo():
    return 0

def f(x):
    return 'Some text ' + str(x) +
        ' some more text ' + str(x) +
        ' even more text that goes on for a while and a while longer, and so on
forever and ever.'

def f(x):
    y = x**2
    return y

    some_string = 'SomeTextThatGoesOnAndOnForSoLongThatItCouldNeverFitOnOneLine'

def boring(args=_None):
    pass

def boring(args=_None):
    pass

def boring(args=_None):
    pass

1 x=~ /foo/

1     def f(x):
2         return x**2

1     def func
2         puts "message"
3     end
4     sike = "sike"

5     def g(x):
6         return 2*x

```

```

1  def (x):
2      return x**2
3  def (x):
4      return 2*x

```

```

1  def hello():
2      print("Hello, world!")

```

```

1  def foo():
2      # This is a comment that contains math:  $\sum_{i=1}^n i$ .
3      return 0

```

```

11  def all(iterable):
12      for i in iterable:
13          if not i:
14              return False
15      return True

```

```

1  public boolean isRowValid(TextField textField, int numRows) {
2      try {
3          int row = Integer.parseInt(textField.getText()) - 1;
4          if (row >= 0 && row < numRows) {
5              return false;
6          } else {
7              System.out.println("Invalid row selection. Please choose a valid
              ↳ row.");
8              Platform.runLater(textField::clear);
9              return true;
10         }
11     } catch (NumberFormatException e) {
12         System.out.println("Invalid input. Please enter a valid integer.");
13         Platform.runLater(textField::clear);
14         return true;
15     }
16 }

```

```

    public static Matrix convertBackToOriginalForm(String[] [] matrix) {
        String[] [] originalFormMatrix = new String[matrix.length] [];
        System.out.println("Matrix length: " + matrix.length);
//        System.out.println("OG Matrix: \n");
        printStringMatrix(matrix);
        for (int i = 0; i < matrix.length; i++) {
            originalFormMatrix[i] = new String[matrix[i].length];
            for (int j = 0; j < matrix[i].length; j++) {
                originalFormMatrix[i][j] = MatrixApp.isFractionMode() ?
                    ↪ convertDecimalToFraction(matrix[i][j]) :
                    ↪ convertFractionToDecimalString(matrix[i][j]);
            }
        }
        System.out.println("Original Form Matrix: \n");
        return new Matrix(originalFormMatrix);
    }

1    public static void printStringMatrix(String[] [] matrix) {
2        for (String[] row : matrix) {
3            System.out.println(Arrays.toString(row));
4        }
5    }

```

```

1    def helloworld():
2        print("Hello, Dracula!")

```