```go
package main

import (
  "errors"
  "fmt"
  "os"
  "strings"
)

func WriteToFile(filePath, stuffToWrite string) {
  f, err := os.OpenFile(filePath, os.O_APPEND|os.O_CREATE|os.O_WRONLY, 0644)
  if err != nil {
    panic(err.Error())
  }
  _, err = f.Write([]byte(stuffToWrite))
  if err != nil {
    panic(err.Error())
  }
}

func Pw(filePath, stuffToWriteAndPrint string) {
  WriteToFile(filePath, stuffToWriteAndPrint)
  fmt.Print(stuffToWriteAndPrint)
}

func DeleteFile(filePath string) {
  _, err := os.Open(filePath)
  if errors.Is(os.ErrNotExist, err) {
    fmt.Printf("No file found to delete on path: %v\n", filePath)
  }

  _, err = os.Stat(filePath)
  if err == nil {
    err = os.Remove(filePath)

    if err != nil && err != os.ErrNotExist {
      panic(err.Error())
```

```
38      }
39      return
40    }
41    fmt.Println("No output file found to delete")
42  }
43
44  func CreateMatrixString(matrix [][]float64) (matrixASString string) {
45    strB := strings.Builder{}
46    //Find out the longest number to print
47    numLen := 0
48    for i := 0; i < len(matrix); i++ {
49      for j := 0; j < len(matrix[i]); j++ {
50        str := fmt.Sprintf("%v", matrix[i][j])
51        if len(str) > numLen {
52          numLen = len(str)
53        }
54      }
55    }
56    //Print each number and fill in gaps to match numLen
57    for i := 0; i < len(matrix); i++ {
58      for j := 0; j < len(matrix[i]); j++ {
59        var numAsStringLen int
60        num := matrix[i][j]
61        numAsStringLen = len(fmt.Sprintf("%v", num))
62        strB.Write([]byte(fmt.Sprintf("%v", num)))
63        if numAsStringLen < numLen {
64          strB.Write([]byte(strings.Repeat(" ", numLen-numAsStringLen)))
65        }
66        if j+1 == len(matrix[i]) {
67          strB.Write([]byte("\n"))
68        } else {
69          strB.Write([]byte("  "))
70        }
71      }
72    }
73    return strB.String()
74  }
```

75