Hacettepe University
Computer Science and Engineering Department
Name and Surname : İbrahim Burak Tanrıkulu
Identity Number : b21827852
Course : BBM203
Experiment : Assignment 1
Subject : Experiments with vectors and matrices
Data Due : 13.11.2019 23.59
Advisors : Alaetin UÇAN
e-mail : b21827852@cs.hacettepe.edu.tr
Main Program : matrixman.c

1. Software Using Documentation

1.1. Software Usage

For run this program, type:
"make" and
"./matrixman [folder_of_arrays] [path_of_command_file] [path_of_output_file]".

1.2 Error Messages

If there is an error in this program, will be shown like "error".

1.3 Input Commands

veczeros [name] [length]
matzeros [name] [x][y]
vecread [filename]
matread [filename]
vecstack [vector1] [vector2] [direction] [name]
matstack [matrix1] [matrix2] [where]
mvstack [matrix] [vector] [where]
pad [matrix] [x] [y] [mode]
padval [matrix] [x] [y] [val]
vecslice [vector] [start] [stop] [name]
matslicecol [matrix] [index] [start] [stop] [name]
matslicerow [matrix] [index] [start] [stop] [name]
matslice [matrix] [y1] [y2] [x1] [x2] [name]
add [matrix1] [matrix2]
multiply [matrix1] [matrix2]
substract [matrix1] [matrix2]

2.Software Design Notes

2.1. Desctiption of the program

2.1.1. Problem

Gaining knowledge on C language, file io, arrays,matrices and dynamic memory allocation.

2.1.2.Solution

1- I created two struct variables which represents vector and matrix.
2- I keeped all vectors and matrices in two struct array.
3- I read input line-by-line and seperated by " " character.
4- I checked conditions and printed "error" when needed.
5- I allocated and placed vectors/matrices to arrays.
6- I writed a function per command and this functions did all operations.
7- I free'd memory that i allocated.

2.2. System Chart

INPUT	PROGRAMS	OUTPUT
<input file>	<my functions>	<output file>

2.3. Main Data Structures

```
struct Vector {
   char* vectorName;
   int* vectorValues;
   int length;
};
struct Matrix {
   char* matrixName;
   int** matrixValues
   int x, y;
};
struct Vector* vectors;
struct Matrix* matrices;
```

2.4. Algorithm

1. Read input file.
   1.1. Find right function to command.
   1.2. Check "error" conditions.
   1.3. Create new vector/matrix or organize vector/matrix
   1.4. (Conditional) Allocate memory on vectors/matrices array.
   1.5. Use function.
      1.5.1. Reallocate memory for operations.
      1.5.2. Do whatever needs.
      1.5.3. Print vector/matrix and (Conditional) return vector/matrix.
   1.6. Free temporary variables.

2. Close input file.
3. Free vectors and matrices.

3.1. Functions Implemented

```
int toInt(char* string);
bool intCheck(char* myString);
void printVec(struct Vector theVector);
void printMat(struct Matrix theMatrix);
void freeVec(struct Vector *theVector);
void freeMat(struct Matrix *theMatrix);
struct Vector veczeros(char* filename, int length);
struct Matrix matzeros(char* filename, int row, int col);
struct Vector vecread(char* filename, char* fileLocation);
struct Matrix matread(char* filename, char* fileLocation);
struct Matrix vecstack(struct Vector vector1, struct Vector vector2, char* direction, char* filename);
void matstack(struct Matrix *matrix1, struct Matrix *matrix2, char where);
void mvstack(struct Matrix *matrix,struct Vector *vector,char where);
void pad(struct Matrix *matrix, int x, int y, char* mode);
void padval(struct Matrix *matrix, int x, int y, int value);
struct Vector vecslice(struct Vector vector, int start, int stop, char* name);
struct Vector matslicecol(struct Matrix matrix, int column, int start, int stop, char* name);
struct Vector matslicerow(struct Matrix matrix, int row, int start, int stop, char* name);
struct Matrix matslice(struct Matrix matrix, int y1, int y2, int x1, int x2, char* name);
void add(struct Matrix *matrix1, struct Matrix *matrix2);
void multiply(struct Matrix *matrix1, struct Matrix *matrix2);
void subtract(struct Matrix *matrix1, struct Matrix *matrix2);
```