Special Characters, Reserved Words, and Functions

Special Characters, Reserved Words, and Functions	Description	Discussed in This Section
`'abc'`	Single quotes enclose a literal character string	2.2.3
`%`	A percent sign indicates a comment in an M-file	2.4.2
`;`	A semicolon suppresses output from assignment statements	2.4.5
`...`	Ellipses continue a MATLAB command to the next line	2.4.5
`=`	The assignment operator assigns a value to a memory location; this is not the same as an equality test	2.2.2
`ans`	The default variable name for results of MATLAB calculations	2.3.1
`clc`	Clears the Command window	2.3.2
`clear`	Clears the Workspace window	2.4.2
`sqrt(x)`	Calculates the square root of `x`	2.4.2

Self Test

Use the following questions to check your understanding of the material in this chapter:

True or False

1. A bag of groceries is an example of abstraction.
2. An algorithm is a series of logical steps that solves one specific problem.
3. It is impossible to write a complete, practical program in any paradigm other than procedural.
4. To be useful to an algorithm, the result of every computation must be assigned to a variable.
5. In programming, if you know the values of `z` and `x` in the expression `z = x + y`, you can derive the value of `y`.
6. Untyped languages are free to ignore the nature of the data in variables.
7. Anything assigned to be the value of a variable is an object.
8. Class is a concept restricted to object-oriented programming.
9. You can permanently save the commands entered in the Command window.

10. Double-clicking an entry in the Command History window lets you rerun that command.
11. You can manually change the values of variables displayed in the Workspace window.
12. You double-click a file name in the Current Directory window to run that script.
13. A Document window lets you view and edit data items.
14. MATLAB permits multiple Figure windows to be open simultaneously.
15. An asterisk on the File Name tab in the Editor window indicates that this is a script that can be executed.
16. MATLAB echoes comments entered in a script in the Command window.
17. When the name of script is typed in the Command window, it will be saved if necessary before it is executed.

Fill in the Blanks

1. Popular programming languages like FORTRAN, C, and MATLAB follow the _____ programming paradigm.
2. _____ languages insist that programmers declare both the name and type of a variable before a value can be assigned to it so that the variable will not be used in an unintended way.
3. The _____ in MATLAB retains a list of all the commands you issued even after a clc command is entered.
4. Unlike other programming languages which indicate the end of a command with a semicolon, MATLAB uses a semicolon to _____.
5. MATLAB works in _____ mode and in _____ mode.
6. The result of any calculation performed in the Command window is saved in the default variable called _____.
7. MATLAB comments start with a _____ sign and are _____ in colour in order to differentiate them from executable commands.
8. The keyboard shortcut keys for adding and removing a MATLAB comment are _____ and _____, respectively.
9. The columns in the Workspace window show the _____of the variable, its _____, and its _____.
10. You _____ the name of a file in the Current Directory window to edit that file.

11. A Document window opens automatically when you _____ a(n) _____ in the Workspace window.
12. Graphics windows are created when a(n) _____ requests a graph.
13. You create comments by putting a(n) _____ in the text file.
14. MATLAB will _____ all text from the comment mark to _____.

Programming Projects

1. Enter two numbers and store them in variables `a` and `b`. Write a script to interchange the values without using a third variable.
2. In the bottom of the ninth inning, the bases are loaded and the Braves are down by three runs. Chipper Jones steps to the plate. Twice he swings and misses. The crowd heads for the exits. The next pitch is a fast ball down the middle. He swings and makes perfect contact with the ball, sending it up at a 45-degree angle toward the fence 400 ft away.
   a. Write a script to determine how fast he must hit the ball to land at the base of the fence, neglecting the air resistance.
   b. Perform a brief experiment to determine whether there was a better angle at which to hit the ball so that it could clear a 12 ft fence.
3. If an ice cream cone is 6 inches tall, and its rim has a diameter of 2 inches, write a script to determine the weight of the ice cream that can fit in the cone, assuming that the ice cream above the cone is a perfect hemisphere. You may neglect the thickness of the cone material. Assume that a gallon of ice cream weighs 8 lb and occupies 7.5 cubic feet.
4. Write a script that validates the relationship between `sin` θ, `cos` θ, and `tan` θ by evaluating these functions at suitably chosen values of θ.
5. I like my shower to remain hot for hours at 100°F, but am too cheap to buy one of those on-demand hot water systems. I don't care how slowly the water runs. The water supply is at 50°F, and the water heater is rated at 50,000 BTU/hour. Write a script to compute the maximum flow rate of my shower (in cubic feet per minute) that keeps the water temperature above 100°F.
6. It takes an average of 45 horsepower to run an electric car at an average speed of 35 mph. Write a script to compute the electrical

Special Characters, Reserved Words, and Functions	Description	Discussed in This Section
`magic(n)`	Generates a magic square	3.5.2
`[v,in] = max(a)`	Finds the maximum value and its position in `a`	3.3.5
`mean(a)`	Computes the average of the elements in `a`	3.3.5
`[v,in] = min(a)`	Finds the minimum value and its position in `a`	3.3.5
`ones(r, c)`	Generates an array filled with the value `1`	3.2.1
`rand(r, c)`	Calculates an `r` × `c` array of evenly distributed random numbers in the range 0...1	3.2.1
`randn(r, c)`	Calculates an `r` × `c` array of normally distributed random numbers	3.2.1
`round(x)`	Rounds `x` to the nearest integer	3.3.5
`size(a)`	Determines the dimensions of an array	3.2.2, 3.5.1
`sum(a)`	Totals the values in `a`	3.3.5

Self Test

Use the following questions to check your understanding of the material in this chapter:

True or False

1. A homogeneous collection must consist entirely of numbers.
2. The function `linspace(...)` can create only vectors, whereas the functions `zeros(...)`, `ones(...)`, and `rand(...)` produce either vectors or arrays of any dimension.
3. The `length(...)` function applied to a column vector gives you the number of rows.
4. You can access any element(s) of an array of any dimension using a single index vector.
5. Mathematical or logical operators are allowed only between two arrays of the same shape (rows and columns).
6. You can access data in a vector `A` with an index vector that is longer than `A`.
7. You can access data in a vector `A` with a logical vector that is longer than `A`.

8. When moving a block of data in the form of specified rows and columns from array A to array B, the shape of the block in A must match the shape of the block in B.

Fill in the Blanks

1. Vector elements have two attributes that make them unique: their _____ and their _____.
2. The operators *, / and ^ are reserved for _____ arithmetic, whereas the operators .*, ./ and .^ are reserved for _____ operations.
3. If an array of size `2*4` is defined as `a = [1 2 3 4;5 6 7 8;]`, _____ and _____ are the values of the operations `max(a)` and `max(max(a))`, respectively.
4. The `length()` function returns the _____ of the array.
5. Arithmetic operations can be performed collectively on the individual components of two arrays as long as both arrays _____or one of them is _____.
6. _____ and _____ are two built-in functions used for creating arrays of any dimension m*n.
7. Removing rows or columns from an array is _____, and can lead to _____. Wherever possible, use _____ to _____.

Programming Projects

1. For these exercises, do not use the direct entry method to construct the vectors. Write a script that does the following:
   a. Construct a vector containing multiples of 3 between 6 and 55, inclusive of the end points. Store your answer in the variable `multiple_three`.
   b. Construct a vector, `evens`, containing even numbers in between 8 and 25, inclusive of the end points.
   c. Construct a vector, `reverse`, containing the numbers starting at 45 and counting backwards by 1 to 35.
   d. Construct a vector, `theta`, containing 16 evenly spaced values between 0 and π.
   e. Construct a vector, `fib`, containing the first 10 Fibonacci numbers. (*Hint:* F(n) = F(n − 1) + F(n − 2) and F(1) = 0, F(2) = 1).
   f. Construct a vector, `random`, containing 15 randomly generated numbers between 1 and 12.

2. Write a script that performs the following exercises on vectors:
   a. You are given a vector `vec`, defined as: `vec = [24 12 33 6 85 43 68 -48 99]`. You decide that you need the even and odd numbers of `vec` separately. Write a script to separate even and odd numbers into two different vectors named `even` and `odd`. Since your commands must work for any vector of any length, you must not use direct entry.
   b. Create a variable called `vecLength` that holds the number of elements in the vector `vec` modified in part a. You should use a built-in function to calculate the value based on the vector itself.
   c. Create variables called `vecSum` and `vecMean` which hold the sum and mean value of the elements in vector `vec`. You should use a built-in function to calculate the value based on the vector itself.
   d. Calculate the sum and average of the values in the vector `vec` without the help of built-in functions. Compare the values with those obtained in part c.
   e. Create a variable called `vecProd` that holds the product of the elements in vector `vec`. You should use a built-in function to calculate the value based on the vector itself.

3. Write a script to solve the following problems using only vector operations:
   a. Assume that you have two vectors named `A1` and `B1` of equal length, and create a vector `C1` that combines `A1` and `B1` such that `C1 = [A1(1) B1(1) A1(2) B1(2) ... . A1(end) B1(end)]`. For example, if `A1 = [2, 4, 8]` and `B1 = [3, 9, 27]`, `C1` should contain `[2, 3, 4, 9, 8, 27]`
   b. Assume that you have two vectors named `A2` and `B2` of different lengths. Create a vector `C2` that combines `A2` and `B2` in a manner similar to part a. However, if you run out of elements in one of the vectors, `C2` also contains the elements remaining from the longer vector. For example, if `A2 = [1, 2, 3, 4, 5, 6]` and `B2 = [10, 20, 30]`, then `C2 = [1, 10, 2, 20, 3, 30, 4, 5, 6]`; if `A2 = [1, 2, 3]` and `B2 = [10, 20, 30, 40, 50]`, then `C2 = [1, 10, 2, 20, 3, 30, 40, 50]`

4. Write a script that, when given a vector of numbers, `nums`, creates a vector `newNums` containing every other element of the original vector, starting with the first element. For example, if `nums = [6 3 56 7 8 9 445 6 7 437 357 5 4 3]`, `newNums` should be `[6 56 8 445 7 357 4]`. *Note:* You must not simply hard-code the numbers into your answer; your script should work with any vector of numbers.

5. You are given a vector, `tests`, of test scores and wish to normalize these scores by computing a new vector, `normTests`, that will contain the test scores on linear scale from 0 to 100. A zero still corresponds to a zero, and the highest test score will correspond to 100. For example, if `tests = [90 45 76 21 85 97 91 84 79 67 76 72 89 95 55]`, `normTests` should be

```
[92.78 46.39 78.35 21.65 87.63 100 93.81 86.6 ...
81.44 69.07 78.35 74.23 91.75 97.94 56.7];
```

6. Write a script that takes a vector of numbers, `A`, and return a new vector `B`, containing the factorial of the positive numbers in `A`. If a particular entry is negative, replace its factorial with 0. For example, if `A = [1 2 -1 5 4 3 -2]`, `B` should be `[1 2 0 120 24 6 0]`.

7. Great news! You have just been selected to appear on Jeopardy this fall. You decide that it might be to your advantage to generate an array representing the values of the questions on the board.
   a. Write a script to generate the matrix `jeopardy` that consists of six columns and five rows. The columns are all identical, but the values of the rows range from 200 to 1,000 in equal increments.
   b. Next, generate the matrix `doubleJeopardy`, which has the same dimensions as `jeopardy` but whose values range from 400 to 2,000.
   c. You've decided to go even one step further and practice for a round that doesn't even exist yet. Generate the matrix `squaredJeopardy` that contains each entry of the original `jeopardy` matrix squared.

8. Write a script named `arrayOperations` that will do the following on the given arrays, and then return a new array of the same size.
   - Add `A` to `B`
   - Subtract `A` from `B`

   The input arguments to your script should be as given below.
   - `A`: a 2D array of any size
   - `B`: another 2D array that has the same size as `A`. If not so, display an error message and the script should terminate execution.

Your script should produce an array, res, of size `M×N` that contains the results of the corresponding operations on `A` and `B`. Test this script by writing another script that repeatedly sets the values of `A`, `B`, `M`, and `N`, and then invokes your `arrayOperations` script.

For example, if `A = [1 2 3; 5 4 6]`, `B = [7 8 9;10 11 12]`, `M = 2`, and `M = 3`, `res` will be `[8 10 12; 15 15 18]`. If `A = [1 2 3;4 5 6]`, `B = [1 2;3 4;5 6]`, then `res` will be an error message `"Incompatible Matrix dimensions"`.

True or False

1. MATLAB keywords are colored green by the editor.
2. Indentation is required in MATLAB to define code blocks.
3. It is possible that no code at all is executed by `if` or `switch` constructs.
4. The word `true` is a valid logical expression.
5. When evaluating a sequence of logical `&&` expressions, MATLAB will stop processing when it finds the first `true` result.
6. The `for` loop repeats the enclosed code block a fixed number of times even if you modify the index variable within the code block.
7. Using a `break` statement is illegal in a `while` loop.
8. The logical expression used in a `while` loop specifies the conditions for exiting the loop.

Fill in the Blanks

1. MATLAB uses _____ in the text to define the extent of code blocks.
2. A logical expression is any collection of constants, variables, and operators whose result is a(n) _____.
3. If you want to continue iterating but omit all further steps of the current iteration, you can use the _____ statement.
4. The _____ command is used to get an input from user.
5. The looping instructions `for()` and `while()` are called _____ looping operations, since the condition is evaluated before the control is passed into the statements inside the loop.
6. If you are in a(n) _____ loop, you can use the `break` statement to skip immediately out of the _____ loop.

Programming Projects

1. Write a script to solve this problem. Assume you have a vector of positive integers named D. Using iteration (`for` and/or `while`) and conditionals (`if` and/or `switch`), separate vector D into four vectors `mulTwo`, `mulThree`, `mulFour`, and `mulFive`.
   - `mulTwo` all of the positive even numbers in D.
   - `mulThree` contains all the multiples of 3 in D.
   - `mulFour` contains all the multiples of 4 in D.
   - `mulFive` contains all the multiples of 5 in D.

For example:

```
if D = [1,8,2,6,3,15],
mulTwo = [2,4,6,8], mulThree = [3,6,15]
mulFour = [8], mulTwo = [15]
```

2. You must use either `for` or `while` to solve the following problems.
   a. Iterate through a vector, `A`, using a `for` loop, and create a new vector, `B`, containing `logical` values. The new vector should contain `true` for positive values and `false` for all other values. For example, if `A = [-300 2 5 -63 4 0 -46]`, the result should be `B = [false true true false true true false]`
   b. Iterate through the vector, `A`, using a `while` loop, and return a new vector, `B`, containing `true` for positive values and `false` for all other values.
   c. Iterate through a logical array, `N`, using a `for` loop, and return a new vector, `M`, containing the value 2 wherever an element of `N` is `true` and the value –1 (not a logical value) wherever `N` is `false`. For example, if `N = [true false false true true false true]`, the result should be `M = [2 -1 -1 2 2 -1 2]`
   d. Iterate through an array, `Z`, using a `while` loop. Replace every element with the number 3 until you reach a number larger than 50. Leave the rest unchanged. For example, if `Z = [4 3 2 5 7 9 0 64 34 43]`, after running your script, `Z = [3 3 3 3 3 3 3 3 34 43]`

3. Your class teacher needs your help. He is preparing the final test scores of all 35 students in his class. He has the `rollNo`, `interimTest1`, `interimTest2`, `quizMark`, and `endExam` scores of all the students. The `interimTest1` and `interimTest2` scores are out of 20 each. The `quizMark` is out of 15 and the `endExam` is out of 45. He needs to calculate the GPAs of all the students. Your class teacher has asked you to write a script that will help him to prepare the Grade chart of all the students. GPAs will be awarded according to the following rules:

`totalMark`	GPA
100-90	S (Excellent)
89-80	A
79-70	B
69-60	C
59-50	D
49-40	E
0-39	F (Fail)

Your script should repeatedly ask for the details of each student and compute the student grade. It should continue until the details of all 35 students are entered.

4. You were just hired for a summer internship with one of the area's best software companies; however, on your first day of work you learn that for the next three months, the only job you will have is to convert binary (base 2) numbers into decimal numbers (base 10). You decide to write a script that will repetitively ask the user for a binary number and return its decimal equivalent until an illegal number (one containing digits other than 0 or 1) is entered. The number entered should contain only the digits 0 and 1. The rightmost digit has the value 2^0 and the digit N places to the left of that has the value 2^N. For example, entering `110101` returns

$$53 = 2^5 + 2^4 + 2^3 + 2^0$$

You must use iteration to solve this problem. Note: The `input (...)` function prompts the user for a value, parses the characters entered according to normal MATLAB rules, and returns the result.

5. You have a friend who has too many clothes to store in his or her tiny wardrobe. Being a good friend, you offer to help to decide whether each piece of clothing is worth saving. You decide to write a script that will compute the value of each piece of clothing. A piece of clothing has five attributes that can be used to determine its value. The attributes are: condition, color, price, number of matches, and comfort. Each attribute will be rated on a scale of 1 to 5. Write a script called `clothes` that will ask the user for the ratings for each attribute and store the result in a vector. The order of attributes in the vector is: `[condition color price matches comfort]`

   The script should compute a value between 0 and 100; 100 represents a good piece of clothing, while 0 represents a bad piece of clothing. The points that should be given for each attribute are shown below:

Condition:	1=>0; 2=>5; 3=>10; 4=>15; 5=>20
Color:	1 => blue => 12;
	2 => red => 2;
	3 => pink => 15;
	4 => yellow => 20;
	5 => white => 12
Price:	1 => 8, 2–3 => 16, 4–5 => 20
Matches:	1–2 => 8, 3–5 => 19
Comfort:	1 => 6, 2–3 => 13, 4–5 =>18

   Note: If a number other than 1–5 is assigned for one of the attributes, no points should be given.

6. Write a function called `divideVector`. It should take in an array of positive and negative integers and return two vectors, `pos` and `neg`,

which store positive integers and negative numbers, respectively. For example,

`divideVector([4 -5 2 1 -7 -3])` should return `[4 2 1]` and `[-5 -7 -3]`

`divideVector([2 -3 4 5 -6 7])` should return `[2 4 5 7]` and `[-3 -6]`

7. Now that you're comfortable with iteration, you're going to have to solve an interesting problem. It seems that the Math department at a rival university has once again dropped the ball, and forgotten the value of `pi`. You are to write a function called `mypi`, which consumes a number that specifies the required accuracy and then approximates the value of `pi` to that accuracy. You are going to use the following algorithm based on geometric probability.

   Think about a quarter circle inside of a unit square (the quarter circle has area $\pi/4$). You pick a random point inside the square. If it is in the quarter circle, you get a "hit"; and if not, you get a "miss." The approximate area of the quarter circle will be given by the number of hits divided by the number of points you chose.

Hint

you could use the function `rand(...)` in this problem.

   Your function should repeat the process of counting hits and misses until at least 10,000 tries have been made, and the successive estimates of `pi` are within the prescribed accuracy. It should return the estimated value of `pi`.

8 Your junior students have to study multiplication tables as part of their curriculum. Unfortunately, they don't know how to generate a multiplication table, given a positive integer less than or equal to 12. You can help them by writing a function which prints this for them. The function should read in any positive number less than 12 and print its multiplication table in the following format.

   For example, if the value entered is 5, then the output should be:

```
1 x 5 = 5

2 x 5 = 10

......

12 x 5 = 60
```

Chapter Summary

This chapter showed you how to encapsulate a code block to allow it to be reused:

- Functions are defined in a file of the same name using the key word `function` to distinguish them from scripts
- Parameters are copied in sequence into the function and given the names of the formal parameters
- Results are returned to the caller by assigning value(s) to the return variable(s)
- Variables within the function can be accessed only in the function's code block unless they are declared global
- Helper functions accessible only to functions within the same file may be added below the main function and otherwise obey the same rules as the main function

Special Characters, Reserved Words, and Functions

Special Characters, Reserved Words, and Functions	Description	Discussed in This Section
`( )`	Used to identify the formal and actual parameters of a function	5.3.2, 5.3.4
`help`	Invokes help utility	5.3.1
`function`	Identifies an m-file as a function	5.3.2
`nargin`	Determines the number of input parameters actually supplied by a function's caller	5.3.4
`nargout`	Determines the number of output parameters actually requested by a function's caller	5.3.4
`global <var>`	Defines the scope of the variable `<var>` as globally accessible	5.3.8

Self Test

Use the following questions to check your understanding of the material in this chapter:

True or False

1. All data used by a function must be passed in as parameters to the function.
2. The name of the first function in an m-file must match the name of the file containing its definition.
3. The first documentation line appears in the Current Directory listing.

4. Functions must consume at least one parameter.
5. The calling code must provide assignments for every result returned from a function.
6. The names of auxiliary functions must begin with `local_`.

Fill in the Blanks

1. The file containing the definition of a function named `function_name` must be _____
2. All the mathematical functions that compute are examples of _____.
3. The list of the names of each data item when a function is defined is called the _____. When this function is called, the caller must provide the same number of data values expected by the function definition, which is known as the _____.
4. _____ describes the situation where the variables within a function are not visible from outside, and the function is unable to cause side effects by making assignments to outside variables.
5. Calling code can only reach the _____ function in an m-file. Other functions in the m-file can only be called from the _____ or _____.

Programming Projects

1. Write a function called `checkPrime` that takes in a number, and checks whether it is prime or not. A number is prime if it has only 1 and itself as its factors. You may assume that number entered is positive. Your function should return a logical value, `true` or `false`.

Hint:

mod(`x, y`) gives the remainder when `x` is divided by `y`.

For example:

```
checkPrime(5) should return true.
checkPrime(24) should return false.
```

2. Write and test the code for the function `mysteryFunction` that consumes a vector, `v`, and produces a new vector, `w`, of the same length where each element of `w` is the sum of the corresponding element in `v` and the previous element of `v`. Consider the previous element of v(1) to be 0.

For example:

```
mysteryFunction( 1:8 ) should return
      [1  3  5  7  9 11 13 15]
mysteryFunction([1:6].^2) should return
      [1  5 13 25 41 61]
```

3. Coming off a respectable 7–6 record last year, your football team is looking to improve on that this season. They have contacted you and asked for your help projecting some of the scenarios for their win–loss record. They want you to write a function called `teamRecord` that takes in two parameters—`wins`, and `losses`, and returns two values—`season` and `wPercentage`. `Season` should be a logical result that is `true` for a winning season. `wPercentage` is the percentage of games won (ranging from 0 to 100).

   For example:

```
[season wPercentage] = teamRecord(3, 9)
should return season = false, wPercentage = 25
[season wPercentage] = teamRecord(10, 2)
should return season = true, wPercentage = 83.3
```

4. Write a function called `classAverage` that takes in an array of numbers and, after normalizing the grades in such a way that the highest corresponds to 100 (see Chapter 3, Problem 5), returns the letter grade of the class average. The grade ranges are as follows:

```
average>90 =>      A
80<=average<90 => B
70<=average<80 => C
60<=average<70 => D
average<60     => F
```

   For example:

```
classAverage( [70 87 95 80 80 78 85 90 66
               89 89 100] ) should return B
classAverage( [50 90 61 82 75 92 81 76 87 41
               31 98] ) should return C
classAverage( [10 10 11 32 53 12 34 74 31 30
               26 22] ) should return F
```

5. Given an array of numbers that could be negative, write a function `NegPos(a)` to calculate and return the sum and average of the positive and negative numbers separately in the single dimensional array, `a`. In order to test your understanding of class concepts, implement the `NegPos(a)` function using iteration. You may not use the built-in functions `sum(...)`, `find(...)`, or `mean(...)` in your solution.

6. Write and test the code for the function `factFun` that allows the user to give a vector `v` of `N` positive integers and produces a new vector, `F`, of the same length where each element of `F` is the factorial of the corresponding element in `v`. For example, `factFunction(1 4 2 8)` should produce `[1 24 2 40320]`. If the user accidentally inputs a negative number, an appropriate message

should be displayed and the function should continue working on the remaining numbers.

7. Write a function called `largest3` that will take in 3 numbers and returns the largest value and an index showing which parameter it was. You may not use the built-in `max()` function.

   For example:

   ```
   largest3(1,3,5) should return 5 and 3
   largest3(8,9,4) should return 9 and 2
   ```

8. Write a function called `sumAndAverage`. It should take in an array of numbers and return the sum and average of the array in that order.

   For example:

   ```
   sumAndAverage([3 2 3 2]) should return 10 and 2.5
   sumAndAverage([5 -5 2 8 0]) should return 10 and 2
   sumAndAverage([]) should return 0 and 0
   ```

9. You are already familiar with the logical operators `&&` (and) and `||` (or), as well as the unary negation operator ~(not). In a weakly typed language such as MATLAB, the binary states `true` and `false` could be equivalently expressed as a `1` or a `0`, respectively. Let us now consider a ternary number system, consisting of the states `true(1)`, `maybe(2)`, and `false(0)`. The truth table for such a system is shown below. Implement the truth table by writing the functions `f=tnot(x)`, `f=tand(x,y)`, and `f=tor(x,y)`. You may not assume that only valid input numbers will be entered.

x	y	tnot(x)	tand(x,y)	tor(x,y)
1	1	0	1	1
1	0	0	0	1
1	2	0	2	1
0	1	1	0	1
0	0	1	0	0
0	2	1	2	0
2	1	2	2	1
2	0	2	2	0
2	2	2	2	2

10. Write a function called `transpose(A)`. This particular function should take in a `N×M` array, `A`, find the transpose of `A`, and store in the array `transA`.

    For example:

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 1 & 4 & 5 & 6 \end{bmatrix};$$

then

$$\texttt{transA} = \begin{bmatrix} 1 & 5 & 1 \\ 2 & 6 & 4 \\ 3 & 7 & 5 \\ 4 & 8 & 6 \end{bmatrix}$$

11. You are playing a game where you roll a die 10 times. If you roll a 5 or 6 seven or more times, you win 2 dollars; four or more times, you win 1 dollar; and if you roll a 5 or 6 three or less times, you win no money. Write a function called `diceGame` that takes in a vector representing the die values and returns the amount of money won.

    For example:

    ```
    diceGame([5 1 4 6 5 5 6 6 5 2]) should return 2
    diceGame([2 4 1 3 6 6 6 4 5 3]) should return 1
    diceGame([1 4 3 2 5 3 4 2 6 5]) should return 0
    ```

    Note: This function should work for any length vector.