NAME: JAMES WASHINGTON

COURSE: DATA STRUCTURES

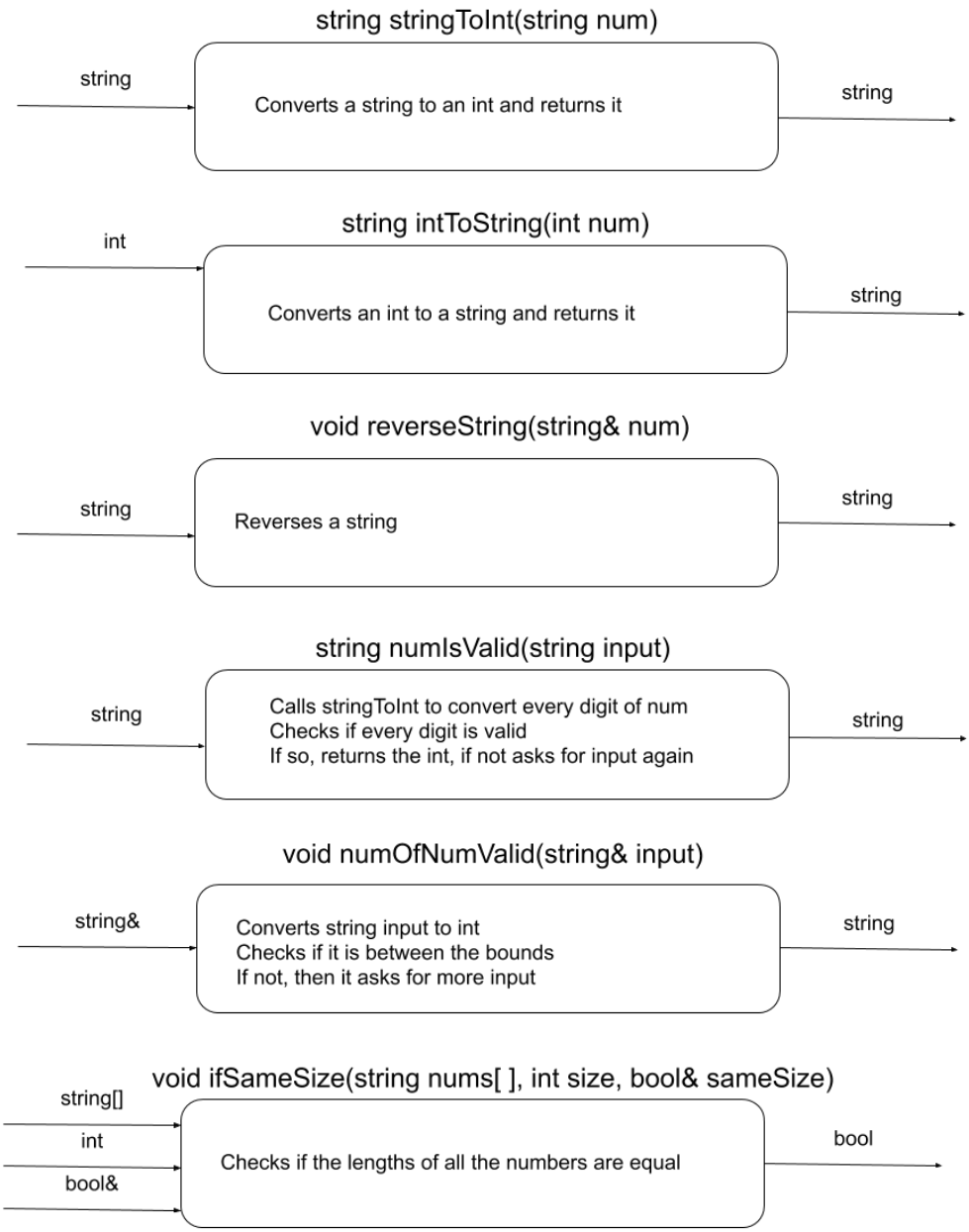
DUE DATE: 2/28/20

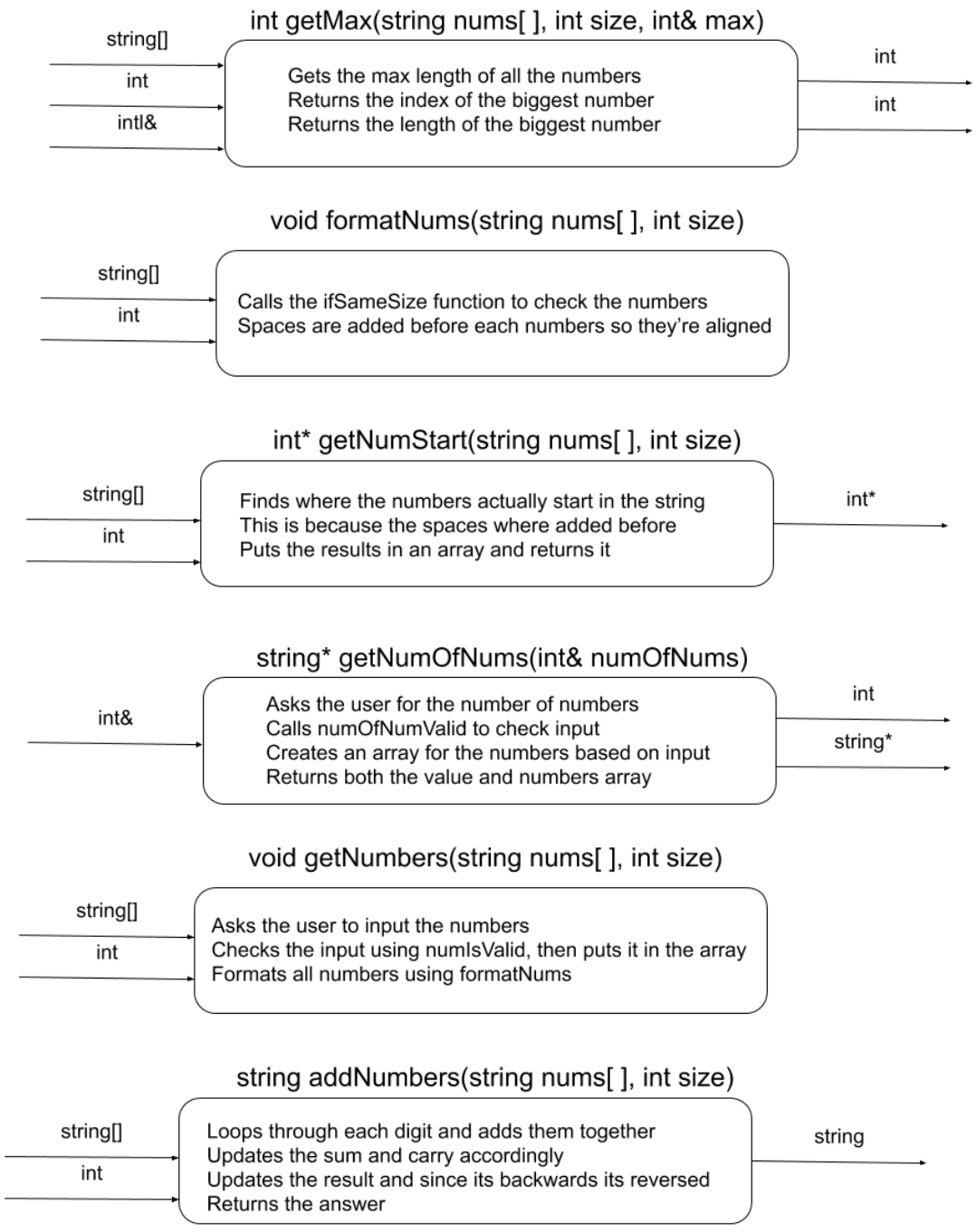
PROFESSOR: DR.NG

Program Design

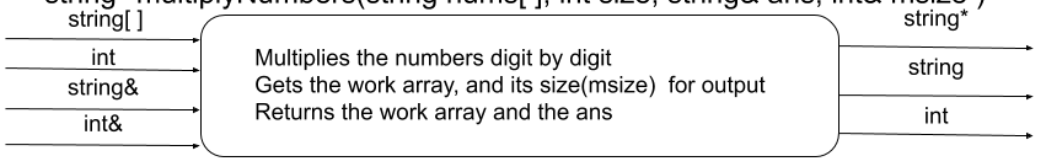
Declares an int to hold the number of numbers the user wants
Declares a string variable to hold the result of the addition
Declares a string variable to hold the result of the multiplication
Declares a size variable for the multiplying results array
Declares a string pointer to contain all the numbers
Calls the getNumOfNums function to get the number of numbers the user wants to enter
Calls the getNumbers function to get all the numbers of the array
Calls the addNumbers function to add all the numbers
Calls the multiplyNumbers function to multiply all the numbers in the array
Calls the outputAdd function to output the addition function
Calls the outputMult function to output the multiplication function

Procedure Specification

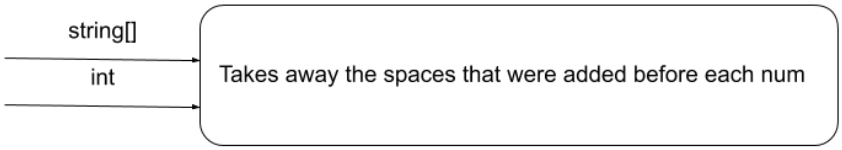



int getMax(string nums[ ], int size, int& max)
string[]
int
intl&
Gets the max length of all the numbers
Returns the index of the biggest number
Returns the length of the biggest number
int
int
void formatNums(string nums[ ], int size)
string[]
int
Calls the ifSameSize function to check the numbers
Spaces are added before each numbers so they're aligned
int* getNumStart(string nums[ ], int size)
string[]
int
Finds where the numbers actually start in the string
This is because the spaces where added before
Puts the results in an array and returns it
int*
string* getNumOfNums(int& numOfNums)
int&
Asks the user for the number of numbers
Calls numOfNumValid to check input
Creates an array for the numbers based on input
Returns both the value and numbers array
int
string*
void getNumbers(string nums[ ], int size)
string[]
int
Asks the user to input the numbers
Checks the input using numIsValid, then puts it in the array
Formats all numbers using formatNums
string addNumbers(string nums[ ], int size)
string[]
int
Loops through each digit and adds them together
Updates the sum and carry accordingly
Updates the result and since its backwards its reversed
Returns the answer
string

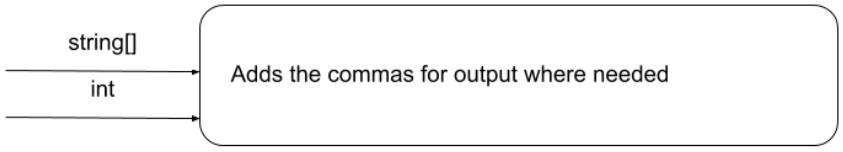
string* multiplyNumbers(string nums[ ], int size, string& ans, int& msize)



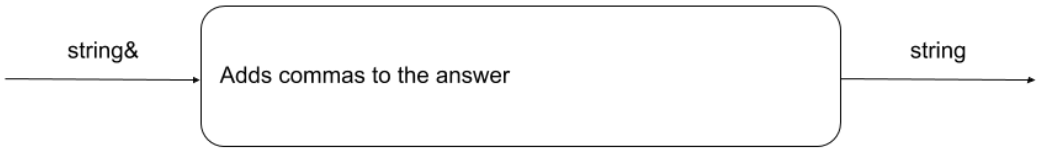
void deformatNumbers(string nums[ ], int size)



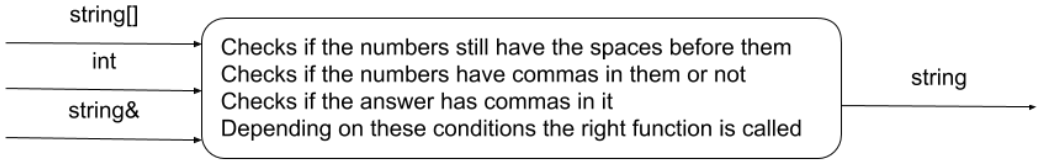
void addCommas(string nums[ ], int size)



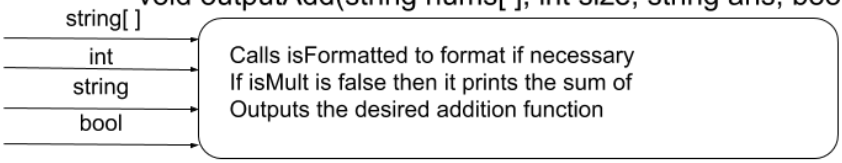
void addCommaAns(string& ans)



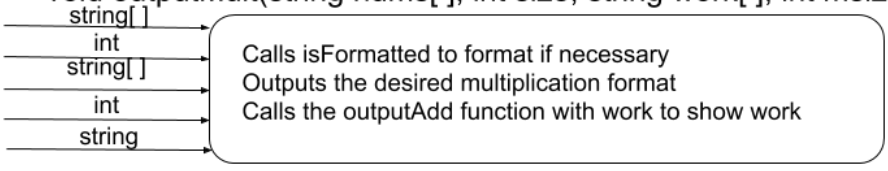
void isFormatted(string nums[ ], int size, string& ans)



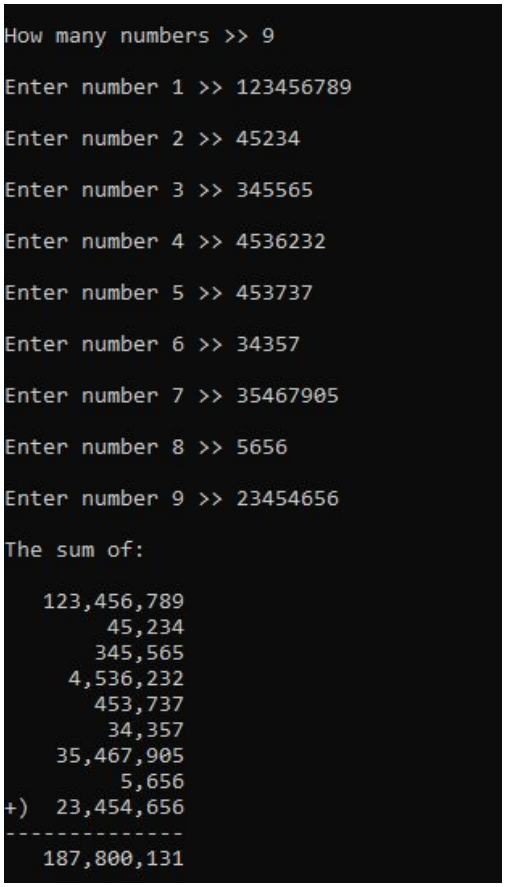
void outputAdd(string nums[ ], int size, string ans, bool isMult)



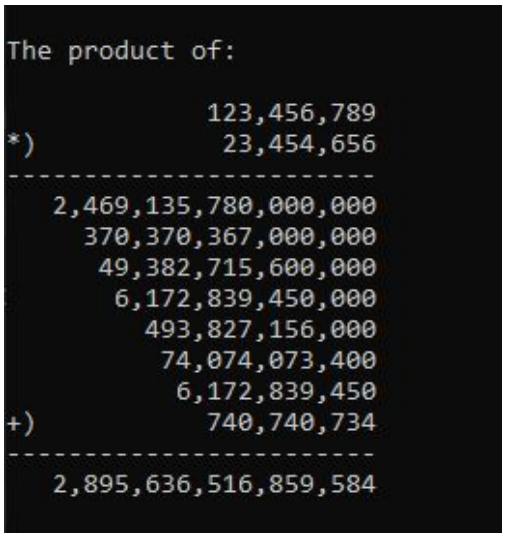
void outputMult(string nums[ ], int size, string work[ ], int msize, string ans)



Program Testing

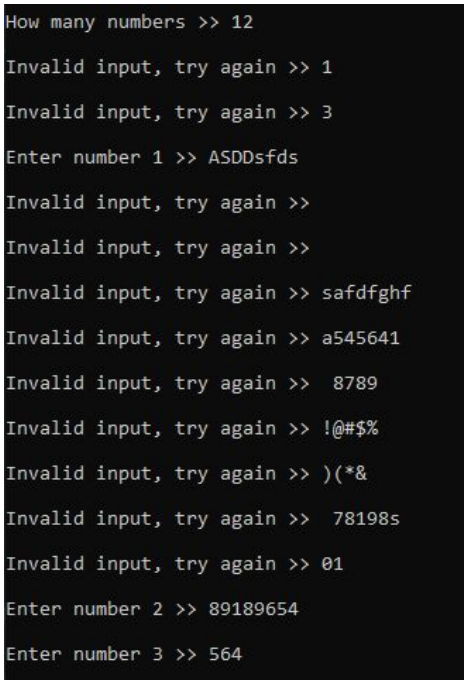


```
How many numbers >> 9

Enter number 1 >> 123456789

Enter number 2 >> 45234

Enter number 3 >> 345565

Enter number 4 >> 4536232

Enter number 5 >> 453737

Enter number 6 >> 34357

Enter number 7 >> 35467905

Enter number 8 >> 5656

Enter number 9 >> 23454656

The sum of:

   123,456,789
        45,234
       345,565
     4,536,232
       453,737
        34,357
    35,467,905
         5,656
+)  23,454,656
--------------
   187,800,131
```

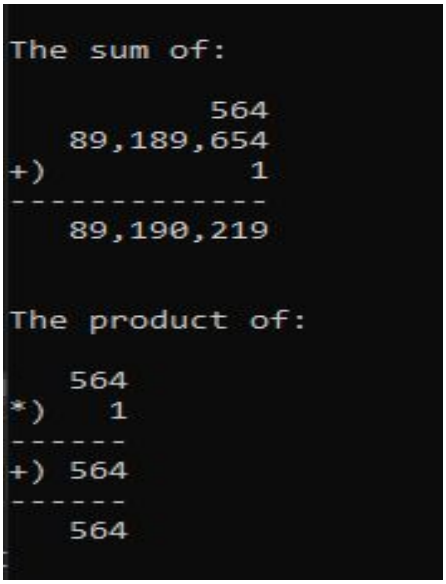


```
The product of:

              123,456,789
*)             23,454,656
------------------------
   2,469,135,780,000,000
     370,370,367,000,000
      49,382,715,600,000
       6,172,839,450,000
         493,827,156,000
          74,074,073,400
           6,172,839,450
+)           740,740,734
------------------------
   2,895,636,516,859,584
```

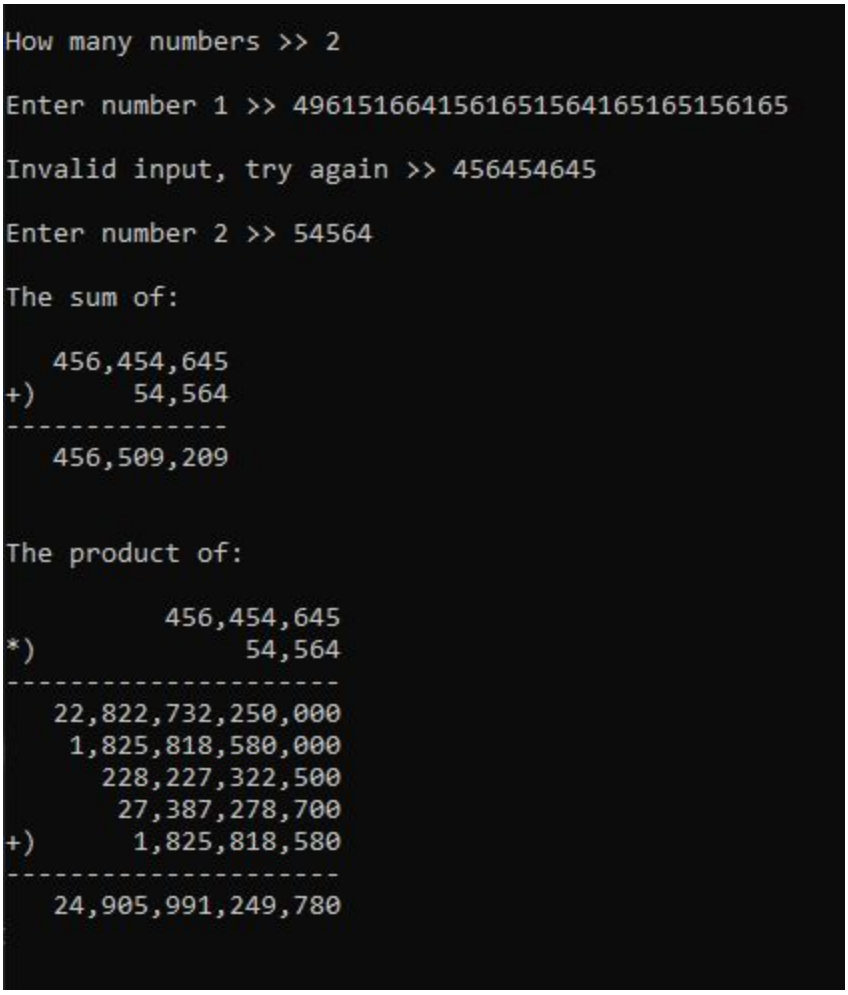
Tested to make sure the output is correct and the numbers are added and multiplied correctly



Checks to make sure input validation is working properly



Checks to make sure math is done correctly



Checks input validation again, and make sure math is done correctly