# CS 5005 Computing Structures – Fall 2019 – Programming Project 1 (100 Points)
# Due September 22nd, 2019 at 11:59 PM

## Objectives

The goal of this project is to create the DataFrame class in C++ as outlined in the project description. Document your project thoroughly as the examples in the textbook (20 point penalty for poor documentation). This includes but not limited to header comments for all classes/methods, explanatory comments for each section of code, meaningful variable and method names, and consistent indentation.

## Project Description

In this project you will implement the DataFrame class along with the all the methods that are represented in the class definition. DataFrame is a table with rows and columns – columns and rows have names associated with them also. For this project we will assume that all the values that are stored in the columns and rows are integers.

After you have tested your class, you have to execute the main program that is also given below.

### *DataFrame Class*

```
#include <iostream>
using namespace std;

class DataFrame {
protected:
      int** table;
      int         noRows, noCols;
      char**      colNames;
      char**      rowNames;
public:
      //Constructors
      DataFrame ();
      DataFrame (int rows, int cols);

      //Output Method
      void display();

      //Setters
      void setRowName(int row, char* name);
      void setColName(int col, char* name);
      int* operator[] (int i); //get row i;

      //Getters
      char** getRowNames();
      char** getColNames();
      int getNumberRows();
      int getNumberCols();
      DataFrame* getColumns(int* columns, int cLen);
      DataFrame* getRows(int* rows, int rLen);
      DataFrame* getRowsCols(int* rows, int rLen, int* cols, int cLen);

      //Destructor
      ~DataFrame();
};
```

*The main function*

```
int main () {

      int c, r;
      int selectC[3];
      int selectR[10];

      // Read the dataframe from input
      // First line: two numbers seperated by space;
      //             first number is the number of rows (r) and
      //             second number is the number of columns (c)

      cin >> r >> c;
      DataFrame* firstDF = new DataFrame(r,c);

      // Second line: strings seperated by a comma (c of them); representing column names
      // Third line: strings seperated by a comma (r of them); representing row names
      // Fourth line and more: c number of integers in each of the r rows (seperated by)
      //    a space between integers in the same row.

      // TODO: Student completes code for the above comment block to get them as input

      // using the display method, print (in the same format as the input):
      // - the column names of the dataframe
      // - the row names of the dataframe
      // - the contents of the table in dataframe

      // TODO: Student completes code for the above comment block

      // Execute the following code

      // Read the column numbers that you want to extract
      for (int i=0; i < 3; i++)
            cin >> selectC[i];

      DataFrame* tempColumns = (*firstDF).getColumns(selectC, 3);

      (*tempColumns).display();

      // Change the row names of select rows

      (*tempColumns).setRowName(2, "Jack");
      (*tempColumns).setRowName(3, "Peter");

      (*tempColumns).display();

      // Read the row numbers that you want to extract

      for (int i=0; i < 10; i++)
            cin >> selectR[i];

      DataFrame* tempRows = (*firstDF).getRows(selectR, 10);

      (*tempRows).display();

      // Change the column names of selected columns

      (*tempRows).setColName(2, "Scores");
      (*tempRows).setColName(3, "Values");

      (*tempRows).display();

      // Extract the rows in SelectR and columns in SelectC
      DataFrame* tempColsRows = (*firstDF).getRowsCols(selectR, 10, selectC, 3);

      (*tempColsRows).display();

      delete tempRows;
      delete tempColumns;
      delete tempColsRows;

   // Sample Code for you and you must execute this

      DataFrame* myTable = new DataFrame(5,5);

      for (int i =0; i < 5; i++) {
            for (int j=0; j < 5; j++) {
                  (*myTable)[i][j] = i*j;
            }
      }

      (*myTable).display();

      delete myTable;
}
```

## *Redirected Input*

Redirected input provides you a way to send a file to the standard input of a program without typing it using the keyboard. To use redirected input in Visual Studio environment, follow these steps: After you have opened or created a new project, on the menu go to project, project properties, expand configuration properties until you see Debugging, on the right you will see a set of options, and in the command arguments type < "path/**input_filename.txt**". The < sign is for redirected input and the **input filename** is the name of the input file (including the path if not in the working directory). A simple program that reads character by character until it reaches enf-of-file can be found below.

```
#include <iostream>

using namespace std;

//The character for end-of-line is ‘\n’ and you can compare c below with this
//character to check if end-of-line is reached.

int main () {

      char c;

      cin.get(c);
      while (!cin.eof()) {
            cout << c;
            cin.get(c);
      }
      return 0;
}
```

## *C String*

A string in the C Programming Language is an array of characters ends with ‘\0’ (NULL) character. The NULL character denotes the end of the C string. For example, you can declare a C string like this:

```
char aCString[9];
```

Then you will be able to store up to 8 characters in this string. You can use **cout** to print out the string and the characters stored in a C string will be displayed one by one until '\0' is reached. Here are some examples:

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | cout result | Length |
|---|---|---|---|---|---|---|---|---|---|---|
| u | s | e | r | n | a | m | e | \0 | username | 8 |
| n | a | m | e | \0 | | | | | name | 4 |
| n | a | m | e | \0 | 1 | 2 | 3 | 4 | name | 4 |
| \0 | | | | | | | | | (nothing) | 0 |

Similarly, you can use a for loop to determine the length of a string (NULL is NOT included). We show this in the following and also show how you can dynamically create a string using a pointer

```
char aCString[] = "This is a C String.";  // you don't need to provide
                                          // the size of the array
                                          // if the content is provided
char* anotherCString;                     // a pointer to an array of
                                          // characters

unsigned int length = 0;

while( aCString[length] != '\0')
{
   length++;
}
// the length of the string is now known

anotherCString = new char[length+1];  // need space for NULL character

// copy the string

for( int i=0; i< length+1; i++)
   anotherCString[i] = aCString[i];

cout << aCString << endl;       // print out the two strings

cout << anotherCSring << endl;

delete [] anotherCString;     // release the memory after use
```

You check http://www.cs.bu.edu/teaching/cpp/string/array-vs-ptr/, other online sources or textbooks to learn more about this.

A sample input file, output file and main program will be uploaded soon. Be on the lookout for that.
Your submission will be on GradeScope where your program will be auto graded. That will be set up a couple of weeks prior to the submission with test cases to pass.

## Constraints

1. In this project, the only header you will use is **#include <iostream>**.
2. None of the projects is a group project. Consulting with other members of this class on programming projects is strictly not allowed and plagiarism charges will be imposed on students who do not follow this.