2025 Spring CPSC 240
Assignment 2 Arrays

Preface

This program has minor application purpose, namely: “Compute the mean of an array of float numbers. The educational purpose is to gain experience programming with arrays. Name both purposes in your internal documentation

Goals

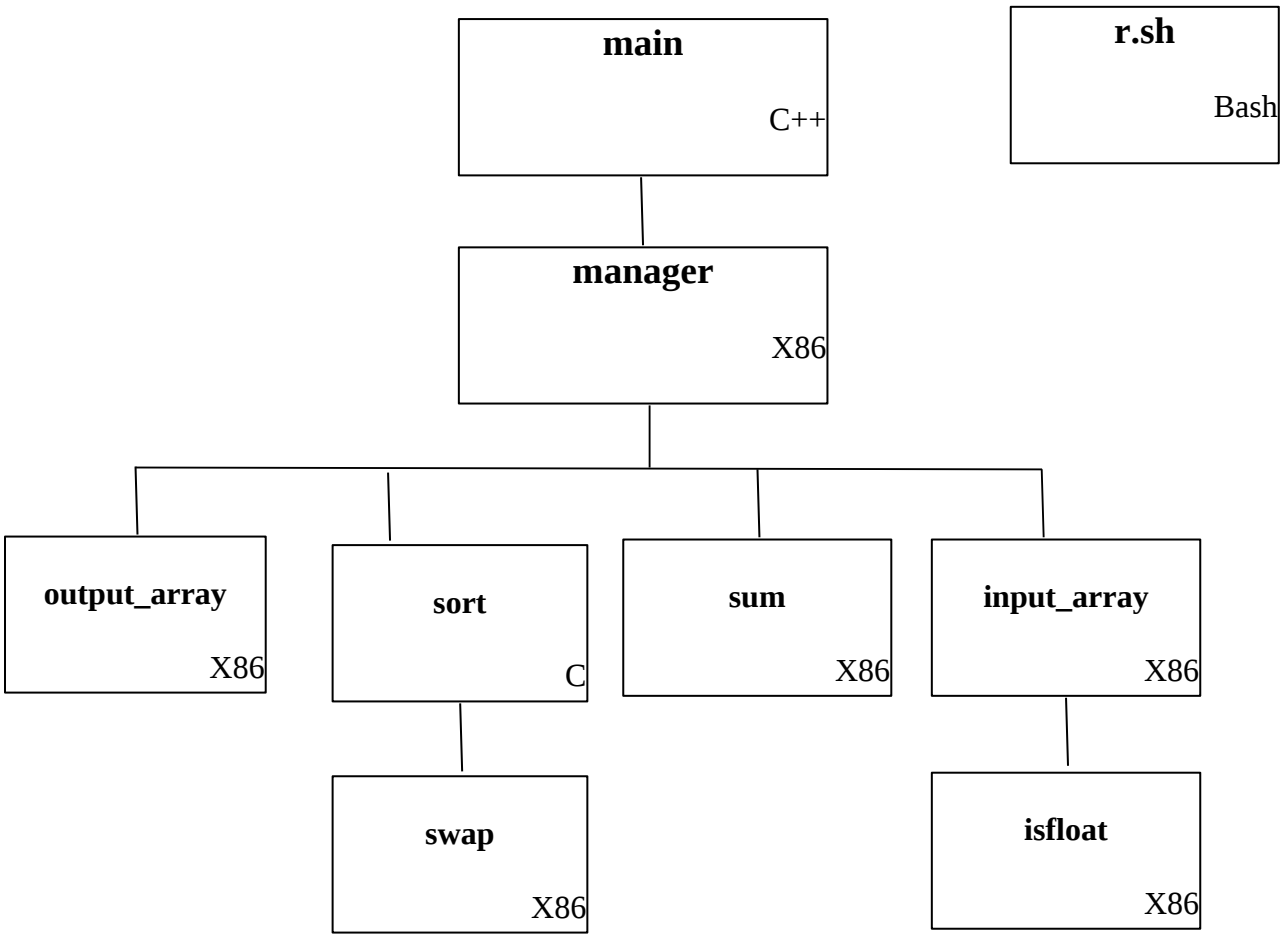
Make an assembly program that teaches all of the following:
- =how to make an array
- =how to implement iteration
- =how to make professional looking programs
- =how to reject invalid inputs
- =add assembly programmer to your list of qualifications in your resume.

Programming requirements

Make a hybrid program that produces the following as shown on a later page of this specification document.. Additional details will be given later in this document.

This style of programming is sometimes called hybrid programming because it uses functions taken directly from the libraries of C and C++. Sometime there is no known function in the C++ library of functions that meets our needs, then we create a brand new C++ function and use it when needed.

Calling diagram



Vertical lines in the diagram are assumed to be pointing down the page. The module in the higher position will call the module in a lower position on this page.

Choose a sort function that you learned in your data structure class and with which you are well familiar. Within that sort function find the place where two floating point values are swapped. In place of swapping the two values inside the sort function, call a function to perform the swapping.

Sample run #1

```
Welcome to Arrays of floating point numbers.
Bought to you by Jaime Green

This program will manage your arrays of 64-bit floats
For the array enter a sequence of 64-bit floats separated by white space.
After the last input press enter followed by Control+D:
  3.2
 -12.25
  9.7744
  21.6
  -4.0003
  13.1  <enter>  <cntl+D>

These numbers were received and placed into an array
3.200000000   -12.25000000   9.774400000   21.666666667   -4.000030002   13.099999999
The sum of the inputted numbers is 16.666666667
The arithmetic mean of the numbers in the array is 2.437148
This is the array after the sort process completed:
-12.25000000   -4.000030002   3.200000000   9.774400000   13.099999999   21.666666667

Main received 16.6666666666, and will keep it for future use.
Main will return 0 to the operating system.   Bye.
============================================================
```

Color codes:
Blue background is produced by module input_array

Yellow background is produced by module manager

Green background is produced by module output_array

Pink background is produced by module Main

The use of “double” implies a 64-bit IEEE number.

The function output_array may output numbers in horizontal rows or in a single vertical column. The programmer may choose.

The function output_array outputs floats with 9 digits on the right side of the point including trailing zeros.

The driver outputs floats with 10 decimal places on the right of the decimal point. Including trailing zeros.

The numbers shown in this document may not be mathematically correct. They are for layout purposes only.

Specifications continued

Change Jaime's name to be your own name.

The calling diagram specifies the programming language of each module.

Generally we follow the guiding principle that each module (source file) contains a single function. This is not a strict rule. If the principal function in a module needs to call another function which is used no-where else in the program then both functions can be placed in the same file. [The words "module" and "source file" are used interchangeably in this discourse.]

Main is not part of the solution. Do not put any code in Main except the display of messages and an occasional number.

The input array function accepts each inputted number as a string, calls the function isfloat to determine if the input was a valid string, and if so then calls atof to conver the string to a valid float number that will be placed in the array.

Isfloat is one of the files of this program. Make sure isfload has a notice of open source license. Submit isfloat along with the other files of this program.

If you modify an open source program with the intent to make it better, then you must add your name and profile information below the names of earlier programmer(s) of isfloat.

Don't submit an isfloat in C or C++.

Sample run #2 including invalid inputs

```
Welcome to Arrays of floating point numbers.
Bought to you by Jaime Green

This program will manage your arrays of 64-bit floats
For the array enter a sequence of 64-bit floats separated by white space.
After the last input press enter followed by Control+D:
  3.2
 -12.25
  9.7744
  36.5W448
   The last input was invalid and not entered into the array.  Try again.
   6.0F2
  The last input was invalid and not entered into the array.  Try again.
  21.6
  -4.0003
  13.1  <enter>  <cntl+D>

These numbers were received and placed into an array
3.200000000   -12.25000000   9.774400000   21.666666667   -4.000030002   13.099999999
The sum of the inputted numbers is 16.666666667
The arithmetic mean of the numbers in the array is 2.437148
This is the array after the sort process completed:
-12.25000000   -4.000030002   3.200000000   9.774400000   13.099999999   21.666666667

Main received 16.6666666666, and will keep it for future use.
Main will return 0 to the operating system.   Bye.

//End of run
```

If the user inputs an invalid number then the user will be asked to try again until a valid float is entered.

When you are done

When your program works correctly. Then make it cosmetically professional. Use a notice of software license in each module. Remove any old comments remaining from the professor's old programs. Remove the professor's name and put your own name in the proper places.

Place descriptive comments before each block within an asm module. The header comment of a block tells the story of what the block does in terms of the application being developed. The application in the program is "Compute the Mean of an Array".

This comment for a block, "Input next number with scanf", is worthless because it tells nothing that can't be read from the source code.

This comment wins: 'Obtain from user the maximum salary for the given pay grade".

Attach the 9 individual files to an email and send it to

tt2025s@proton.me if your 240 class meets on Tuesdays and Thursdays

mw2025s@proton.me if your 240 class meets on Mondays and Wednesdays.

In the subject line place the string "CPSC 240-W Assignment 2" where W is your section number.

The word is "attach". Do not even think about placing your valuable program in cloud storage.

Don't email me broken code: segfaults, missing modules, non-compiling modules, non-assembling modules, missing bash files, etc. These are non-executing programs.

Be sure to include a bash file. It is part of the program. Make sure that every line of the bash file terminates with a LF (line feed = ascii value 10). Text files made in windows do not generally have LF terminators.

Due date & time: February 22, 2025 at 11:59pm

Points: All assigned programs in the course are worth 20 points.

Testing your submitted program

There are two grader persons. Both have asked to remain anonymous. For each section I usually grade about 15% of the submitted programs. A grader evaluates the rest of the submissions.

We all have to be grateful for graders. I am grateful for graders. Without them this would be a class with no assignments. There would only be a concept midterm and a concept final, which is not enough to get you hired in any company.

When a grader completes the work of grading an assignment then he will send me a list showing names and scores. I will replace each student name with the last 4 digits of that student's cwid. Then I send the list to one of the SI instructors who will post it in Discord for everyone to view it.

//End of Assignment 2 Requirements Document.

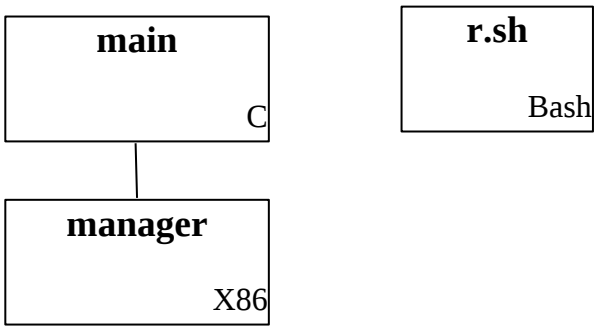
To reader: This appendix was written about 2 years ago. It contains only suggests for starting the construction of this program. You decide if you want to apply the techniques suggested here.

Appendix: Incremental Construction of a Program

The proceeding pages of this document describe a complete program. The instructor of this course recommends incremental development as you advance toward completion of the entire program. Caveat: in software engineering classes you will learn other strategies for building and developing software. In those classes use the techniques presented in class lectures. Here incremental development is recommended, but not required.

Phases of development of Assignment 2 program.

Phase 1

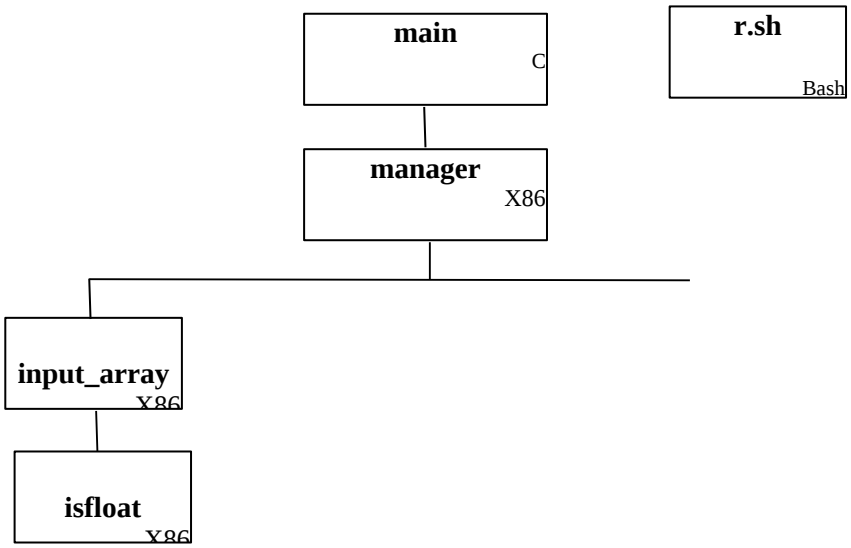


Phase 1 output

```
Welcome to Arrays of floating point numbers.
Bought to you by Jaime Green
This program will manage your arrays of 64-bit floats
For the array enter a sequence of 64-bit floats separated by white space.
After the last input press enter followed by Control+D:
These numbers were received and placed into an array
The variance of the inputted numbers is 0.000000
Main received 0.0000000, and will keep it for future use.
Main will return 0 to the operating system.   Bye.
```

//Make this partial program work correctly before moving on to the next phase.

Phase 2



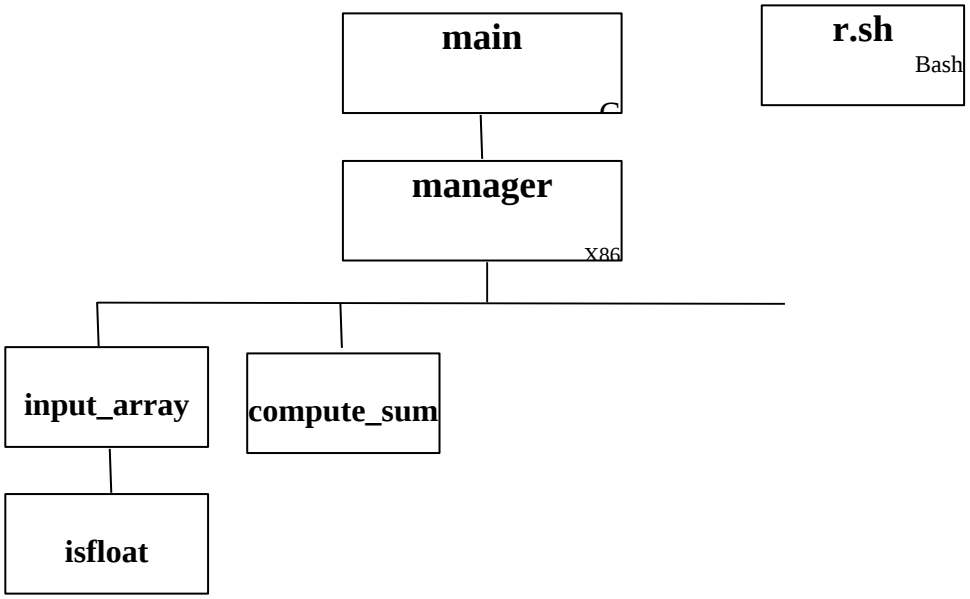
Make these modules work correctly before moving to the next phase.

Welcome to Arrays of floating point numbers.
Bought to you by Jaime Green

This program will manage your arrays of 64-bit floats
For the array enter a sequence of 64-bit floats separated by white space.
After the last input press enter followed by Control+D:
  3.2
 -12.25
  9.7745
  21.6
  -4.0003
13.1 <enter> <cntl+D>
These numbers were received and placed into an array
The variance of the inputted numbers is 0.000000
Main received 0.0000000000, and will keep it for future use.
Main will return 0 to the operating system. Bye.

Make sure this stage of program development runs perfectly before moving on.

Phase 3

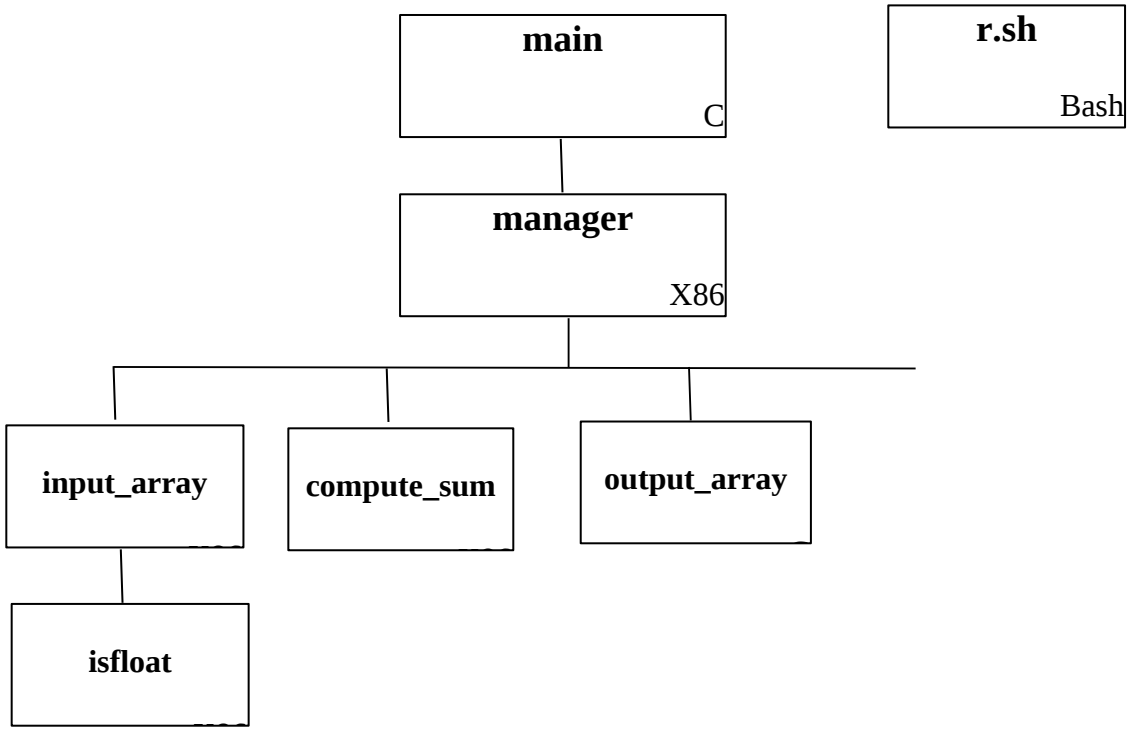


Output

Welcome to Arrays of floating point numbers.
Bought to you by Jaime Green
This program will manage your arrays of 64-bit floats
For the array enter a sequence of 64-bit floats separated by white space.
After the last input press enter followed by Control+D:
  3.2
 -12.25
  9.7745
  21.6
  -4.0003
  13.1 <enter> <cntl+D>
These numbers were received and placed into an array
The mean of the numbers in the array is 22.657444
The variance of the inputted numbers is 16.666666
Main received 16.6666666666, and will keep it for future use.
Main will return 0 to the operating system. Bye.

Make sure Phase 3 runs perfectly before going to the next phase

Phase 4



Output

Welcome to Arrays of floating point numbers.
Bought to you by Jaime Green

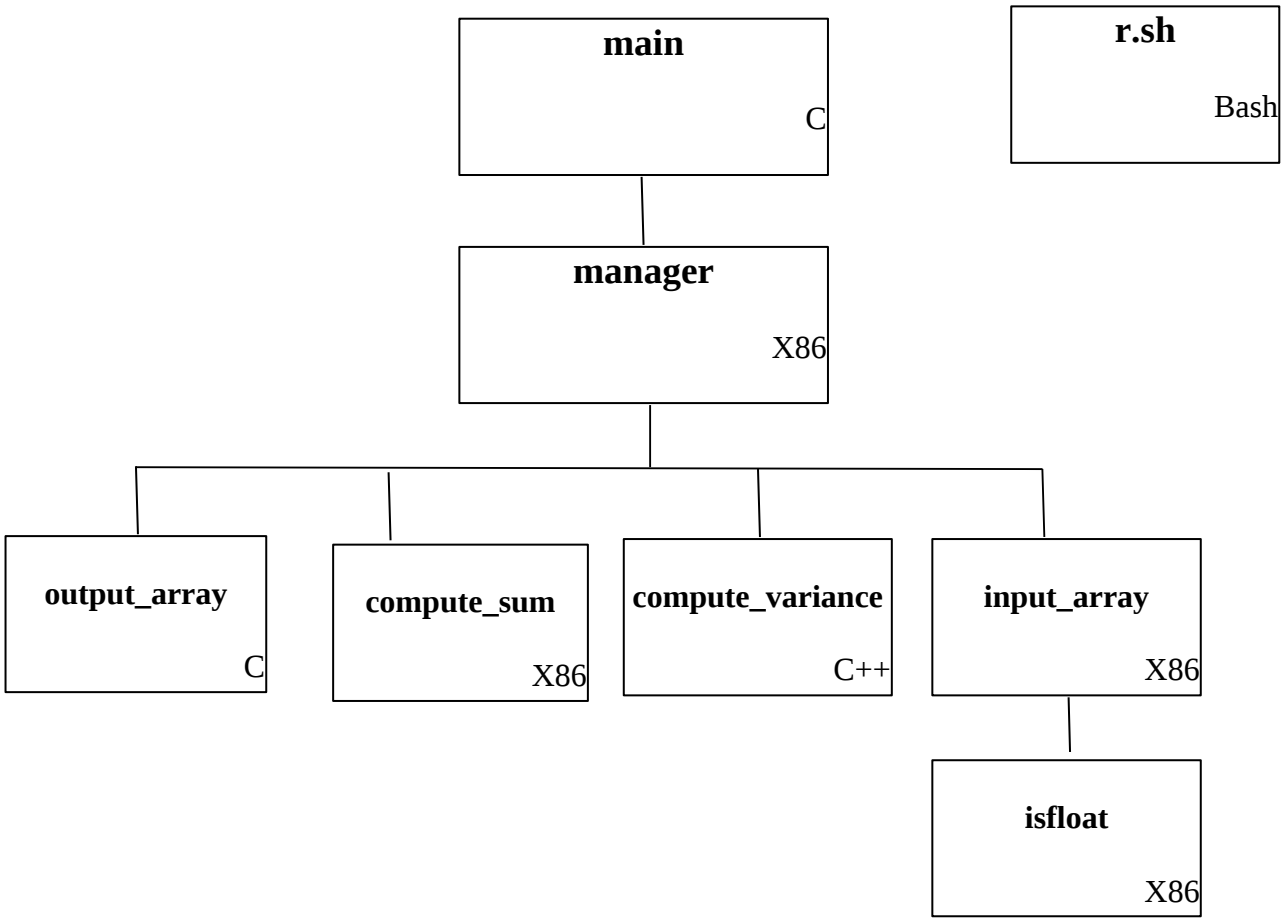
This program will manage your arrays of 64-bit floats
For the array enter a sequence of 64-bit floats separated by white space.
After the last input press enter followed by Control+D:
  3.2
 -12.25
  9.7745
  21.6
  -4.0003
  13.1 <enter> <cntl+D>

These numbers were received and placed into an array
3.19999 -12.25000 9.77450 21.60000 -4 .00029 13.09992
The mean of the numbers in the array is 22.657444

Main received 16.6666666666, and will keep it for future use.
Main will return 0 to the operating system. Bye.

Be sure that the program performs perfectly before going to the next phase.

Phase 5 (Final phase) Add the variance function



This concludes the final stage of development. Make sure the program runs correctly with both valid and and invalid inputs

There is a serious benefit derived from incremental program development in this class. That benefit is that the developer may halt development at any of the phases and have at hand a partially working program worth substantial partial credit. That is a very good outcome.

End of Appendix