Department of Computer Science
University of Pretoria

Programming Languages
COS 333

Practical 5: Fortran and COBOL

October 4, 2022

1 Objectives

This practical aims to achieve the following general learning objective:

- To gain and consolidate some experience writing programs in several different imperative programming languages, including: Fortran and COBOL;
- To consolidate a variety of basic concepts related to imperative programming languages, as presented in the prescribed textbook for this course.

2 Plagiarism Policy

The Department of Computer Science considers plagiarism as a serious offence. Disciplinary action will be taken against students who commit plagiarism. Plagiarism includes copying someone else's work without consent, copying a friend's work (even with consent) and copying material (such as text or program code) from the Internet. Copying will not be tolerated in this course. For a formal definition of plagiarism, the student is referred to `http://www.ais.up.ac.za/plagiarism/index.htm` (from the main page of the University of Pretoria site, follow the `Library` quick link, and then click the `Plagiarism` link). If you have any form of question regarding this, please consult the lecturer, to avoid any misunderstanding. Also note that the principle of code re-use does not mean that you should copy and adapt code to suit your solution. Note that all assignments submitted for this module implicitly agree to this plagiarism policy, and declare that the submitted work is the student's own work. Assignments will be submitted to a variety of plagiarism checks. Any typed assignment may be checked using the Turnitin system. After plagiarism checking, assignments will not be permanently stored on the Turnitin database.

3 Submission Instructions

Upload your practical-related source code files to the appropriate assignment upload slots on the ClickUP course page. For your Fortran submission you must implement and submit a single file named `s99999999.f`, where `99999999` is your student number. For your COBOL submission you must implement and submit a single file named `s99999999.cob`, where `99999999` is your student number. Multiple uploads are allowed, but only the last one will be marked. The submission deadline is **Tuesday, 11 October 2022, at 23:00**. Include comments in your source code files, which explain how to compile and execute your programs.

4 Background Information

For this practical, you will be writing programs in Fortran 2008 and COBOL. You will have to compare these languages in terms of their support for different concepts related to data types, control structures and subprograms. To do this, you will have to write short programs to demonstrate how each language handles the

concept under consideration. These programs will not be long, but will demonstrate the concept adequately, and should allow you to understand the language's support (or lack of support) for the features in question.

Your Fortran 2008 program should compile using version 7.5.0 of the `gfortran` compiler, while your COBOL program should compile using version 1.1.0 of the OpenCOBOL compiler. Support for Fortran 2008 is provided under Linux as part of the GNU Compiler Collection with the `gcc` compiler. OpenCOBOL is not a strictly standard conforming implementation of COBOL, but implements most of the COBOL 85 and COBOL 2002 standards, as well as some proposed features from the COBOL 2014 specification. If you decide to use a different compiler for any language, **make sure that you test your programs using the specified versions of these compilers**.

The course ClickUP page contains documentation related to the Fortran language [1] and the `gfortran` compiler [3]. Also included is documentation related to the OpenCOBOL compiler [2].

Note that that your Fortran program code must comply with the `gfortran` compiler's fixed format. Your COBOL program must also compile with COBOL's fixed format source file specification. Refer to the documentation for each compiler for information on how to enforce fixed format source files. Adherence to the fixed format will contribute to your mark for both your submissions.

5 Practical Tasks

You have to implement the same simple statistical measure program in Fortran 2008 and COBOL. The program reads in and stores a sequence of integer values and then prints out statistics related to these values. Note that in the following discussion function signatures are provided in C-like syntax, and are intended only to illustrate the basic nature of the return and parameter types. Your programs must adhere to the following requirements:

- You do not need to define any user-specified data types. Simply use an array to store your data.
- You must define a subprogram with the signature `void readData(arr)`, where `arr` is an array containing five integer values. The `readData` subprogram should prompt the user for integer inputs and populate the array with these values. The array should be modified by reference, meaning that there should be no return value for `readData`. You may assume that only five data values will be entered.
- Declare a final subprogram with the signature `float stdDev(arr)`, which receives an array of floating point values (here called `arr`) and returns the sample standard deviation of the values in the array. The sample standard deviation of a set of data values is computed as follows:

$$S_N = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \overline{x})^2}$$

  where $\{x_1, x_2, \ldots x_N\}$ are the floating point vaues in the array, N is the number of values contained in the array, and $\overline{x}$ is the mean value of the floating point values in the array.
- Finally, your program should prompt the user for values that will be stored in the array by calling the `readData` subprogram. Your program should then determine the sample standard deviation of the values in the array using the `stdDev` subprogram, and print out this value.

Note that in the above description the term "subprogram" refers to a unit of execution (you will probably know such units of execution as functions or methods). Some programming languages support different kinds of subprograms, and you should use the most appropriate type of subprogram for the task at hand.

It is also possible that language limitations may affect aspects of the functions in one way or another (for example, in terms of the return types allowed from subprograms). Should a subprogram not support a feature you would typically use, you must find another way to provide the required functionality. For example, if certain data types cannot be returned by a subprogram, consider whether you can pass a variable of the required type to the subprogram by reference. Similarly, if a value cannot be passed by reference, consider whether it can be returned instead.

6 Marking

Each of the implementations will count 5 marks for a total of 10 marks. Both the implementation and the correct execution of the programs will be taken into account. Your program code will be marked offline by the teaching assistants. Make sure that you upload your complete program code to the appropriate assignment upload slot, and include comments at the top of your program source file, explaining how your program should be executed, as well as any special requirements that your program has. Do not upload any additional files other than your source code.

References

[1] PGI Compilers and Tools. Fortran reference guide: Version 2017, 2017.

[2] Gary Cutler. OpenCOBOL 1.1 programmer's guide, September 2010.

[3] The `gfortran` team. Using GNU Fortran: For GCC version 6.3.0, 2016.