Re-engineering Process of tictactoe

Synopsis :

The code for tic tac toe was given to us, but it was in fortran 77. The first step taken in the process of re-engineering was to analysis the program. In order to do an analysis of this program spending sometime on reviewing the program structure, fortran 77 syntax and the flow of the program to have a better understanding of the program. With understanding of on the flow of the program and syntax a flowchart can determined. Having the flowchart it became an essential tool in determining the logical process of the code and to understand the underlying functionality.Using the flowchart provided eased up process of re-engineering the unstructured code in form of go-to statements.Some of the comments provided in the code also helped visualizing the layout of what is happening in the program.

The second step was to go through the code and fix and restructure the easy pars before proceeding into complex structure such as go-to statements. The if statement were changed from fortran 77 syntax to fortran 95, basically the structure was to change it to if/end if statments. Adding more comments to have more understanding of the code,changing the variable name to something that makes sense and the entire program is converted from uppercase to lowercase to make it more legible to read and maintain.The do loops in fortran 77 had to be changed to standard fortran 95 syntax.Go to statements and modularity were the last thing to change since it involved in restructuring a huge chunk of code.

When changes were made to the code, each changes that was done to re-engineer the legacy code was saved and then that copy was emailed to me and also backed up on a my dropbox so that I will never lose a working copy of the code. I consistently made sure that it worked on thornbrough lab computer such that I had a working copy of the code. Insteaded of making new copy for every changed I made on

the code I ill work on the same code but when I get little further from where I was before I will immediately save that and have a backup of it on dropbox so if in anycase I was about loose the original copy I always had a backup of the program.

The design decisions, I choose to study and understand the fortran 77 using the online guides and notes provided by the professor. After knowing what fortran 77 can do I chose to study the program tic tac toe so I can get a better idea of the functionality. Then I focused on the easy structures of the code and would try to resolve any compilation errors one by one. Then I would work up till everything was restructured and met the specifications that were outlined on the assignment.

Legacy structures :

1. The first legacy structure that I encountered was comments in fortran 77. The comments were specified by using the “!” examination mark in fortran, which I have never encountered since in C or java the comment will be specified using “/**/”. In fortran 95 it didn't need a lot of change since the comment from 77 and 95 uses the same syntax.

2. The second legacy feature that was encountered was the lack of a program structure. In order to meet the fortran 95 standards, program structure was changed by adding a program header, the name of the program in the beginning of the code (seen in line 21) and an end program statement at the very end (line 66).

3. Identifiers used in the program were also enhanced to be more legible to read, understand and maintain the legacy code. These involved changing the variables names into something that makes more sense (seen in line 25,26).

4. Fixed format from the legacy code was removed to meet the free format standards of fortran 95 by adding whitespace, indentation, paragraphing and changing variables names as discussed above as well as converting from uppercase to lowercase.

5. Declarations statements were also changed to meet the fortran 95 standards. For example in fortran 77 Integer would be declared as “Integer move” but this has been modified by adding the double semi colons before the variable name such as “Integer :: move” as (seen in line 26).

6. Code was also organized by adding more subroutine and comments were added where appropriate to give a better understanding of the program.

7. Input and output prompts were also changed and new messages have been added to user (example seen in line 61) to provide proper feedback and to make the legacy code more user friendly.

8. Legacy if statements were restructured to meet the fortran 95 standards. If statements with go-to statements followed were removed and restructured. Also, the structure of the if statements were changed to if, then, end if to meet the fortran 95 standards (examples seen in line 41-63) and any other arithmetic if statements have been removed.

9. Arrays and variables have been converted to also meet the fortran 95 standards by changing their declaration statements by adding the dimension attribute providing the length of the arrays instead of the operator “Integer*1” used in the fortran 77. When using a two-dimensional array I had to use “reshape” function in order to set the variable to those arrays, in fortran 77 they set the array variable using data which gave me a compiling error in fortran 95.

10. Loop structures have also been reengineered. Instead of using do- continue statements as done in the legacy code, they were restructured to use do-end do statements removing the need of labels, and implicitly declaring the looping index (seen in line 337-349).

11. The ".EQ.", ".GE." logical operators from fortran 77 used in the if statements have also restructured to meet the fortran 95 standards by replacing them with "==" etc as seen in lines 45 & 73.

12. Go-to statements have been removed and suppressed by restructuring the entire program with if, else if, else statements to avoid the use of labels and jumping of the program code. This was done by following program's flow and determining where the program jumps and replacing it with in if statements to avoid the jumps. This can be seen through lines 41-63 and lines 388-396.

Questions and answers :

1. Would it have been easier to translate the program into a language such as C?

   It would be have been easier to rewrite the program from scratch in a language such as C because the complexity and the length of the program is not too so much and could have been more maintainable. It would have taken a lot of re-engineering since the code for fortran and C are not the same, but it would have made easier since C is common used language and the syntax for it is everywhere in online. In C it would have made the program much more readable than fortran 77. If the given program had about 10,000 lines of code then using C to re-engineering would be a pain since converting all the lines of code to C syntax will take lot of time whereas re-engineering to Fortran 95 would have been much more easier.

2. Is there a better way of modernizing the program?

   I believe on what's given to us and what we did for this tic tac toe program is all the modernizing that we can do. Since the assignment requirements were convert fortran 77 code fortran 95 to show how efficient and good fortran 95 is, and I don't think you can modernize the tic tac toe any more.

3. Given your knowledge of programming, was Fortran easy to learn?

   Fortran was easier than many other language that I had to learn.It had a simplestic syntax and structure compare to Java or C. The goto statement was a bit confusing since it's a logical jump call, where when I starting to understand the flow of the program the goto statement will randomly be their and confuse me so I have to go back and forth the same code to understand what the goto statement doing. Other than that every other syntax and structure were pretty to understand.

4. What structures made Fortran usable? (In comparison to C for instance)

   The fortran structures arrays are one of the interesting that this language had compared to C since it provides some function to be called on the array constructs an array of a specified shape from a template array the reshape function will do that. Fortran language also had subroutines that don't need to defined at the top of the file which is an improvement over C. Commenting was bit better in fortran compared to C because you don't have to “/*comments */” do like that and can just add comment using just one symbol “!”.