Mainframe Development 1 - LAB 6

The emphasis in this lab is the use of structure charts to assist in designing your programs and further use of PERFORM (with PERFORMS within PERFORMED paragraphs). When reading through the requirements and planning functionality,   
be careful in decomposing tasks, so that functionality is implemented in the most appropriate paragraph.

This program is to print out a report of employees and salary increases.   
Download the sample output and data file from the DC Connect Dropbox.

# Input

The input data file has the following layout:

EMPLOYEE NAME EDUCATION YEARS PRESENT

NUMBER CODE SERVICE SALARY

1-3 4 - 18 19 20-21 22-28

004 Archer Bill N 01 5000.00

185 Donneman Tom G 06 12000.00

**Note**:   
The EDUCATION CODE will contain either a ‘G’ for employees who are graduates,  
or ‘N’ for non-graduates.   
These are the only two values that will appear; there will NOT be erroneous data.

# Requirements

1. The program is to classify all employees, both GRADS and NON GRADS, as follows. These classifications (i.e. SEN PROG, etc.) along with associated values **are to be set up as constants** in WORKING-STORAGE.
   1. A graduate employed more than 15 years is an analyst   
      (print out ANALYST).
   2. A graduate employed 7 to 15 years inclusive is a Senior Programmer   
      (print out SEN PROG).
   3. A non-graduate employed more than 10 years is a programmer   
      (PROG).
   4. A graduate employed less than 7 years but more than 2 is also a programmer (PROG).
   5. A non-graduate employed 10 years or less but more than 4 years is a   
      junior programmer (JR PROG).
   6. A non-graduate employed 4 years or less is unclassified.   
      (Leave blank, use Move Spaces, NOT a WS literal).
   7. A graduate employed 2 years or less is unclassified   
      (leave blank, use Move Spaces).

**DESIGN REQUIREMENT:** Use a structure chart to help with the process of designing your program. Break your solution down into at least 5 tasks (paragraphs), one for each classification, to process the requirements for the classification, to be used by both the GRADUATES/NON GRADUATES depending on their years of service. Then, within the main IF statements, just PERFORM those paragraphs as required. You may also want to consider having one paragraph each for Grads and Non-grads themselves.

1. The program is to calculate pay increases as follows:  
     
   Analyst 11.8% These pay increase %  
   Senior Prog 8.3% **are** to beProgrammer 6.7% set up in   
   Junior Prog 4.2% WORKING-STORAGE   
   Unclassified 0% as program constants.
2. The program is to calculate and print out the average increase (2 places of decimal) for each of the 4 classifications. Print with SHORT TITLES on 1 or 2 lines.   
   Use Z and comma editing.
3. The output report is to have the main heading EMPLOYEE SALARY REPORT and suitable column headings; line spacing is your choice.   
   My attached sample shown is just a model, and you do not have to follow it exactly.
4. For each employee print the employee number, name, years, classification,   
   (i.e. SEN PROG, etc.), present salary, % increase **with a % sign**, pay increase   
   and new salary.
5. Print 15 lines on each page,   
   and include a PAGE NUMBER as shown on the sample output.
6. For the employees on each page:
   1. Count the employees that fall into each class
   2. Display the total count at the bottom of each page   
      as displayed in the sample output.   
      **NOTE**: This should only include the employees   
      that are listed on the page in question.
7. Print the pay increase amount with a floating $, a comma and a + sign   
   on the right hand end, i.e. $2,610.01+.
8. For UNCLASSIFIED leave the % increase blank (spaces)   
   and print the pay increase amount as $0.00+.   
   Don’t print the %, of course.
9. Print the NEW SALARY with a fixed $, a comma and Z's, i.e. $ 27,251.17.
10. Print the PRESENT SALARY with Z's, and a comma. No $.
11. You MUST create your OUTPUT DETAIL LINE in WORKING-STORAGE,   
    NOT within the FD record layout.   
    You will be using FROM consistently in all WRITE statements.

# SUBMISSION:

A zip or 7z archive containing your entire visual studio project and .sln file.

A copy of your structure chart

# due: April 6th @ 11pm