1.

Reading files

COBOL is better for manipulating files since it declares rigid record in the beginning, making file read/write in specific format easy.
FORTRAN, on the other hand, could also read/write specific file format easily. However, if the given records are cross-referencing much(have to use many record of other files), then FORTRAN has to use parameters passing/common variable to do the trick. In which case COBOL's global variable gives cleaner and higher readability code.

```
01 EMPLOYEE-RECORD.
    05 STAFF-NUMBER PIC 9(4).
    05 EMPLOYEE-NAME.
        10 EMPLOYEE-LAST-NAME PIC X(10).
        10 EMPLOYEE-FIRST-NAME PIC X(20)
    05 GENDER PIC A.
    05 DATE-OF-BIRTH.
        10 YYYY PIC 9999.
        10 FILLER PIC X VALUE '-'.
        10 MM PIC 99.
        10 FILLER PIC X VALUE '-'.
        10 DD PIC 99.
    05 HIRING-DATE.
        10 YYYY PIC 9999.
        10 FILLER PIC X VALUE '-'.
        10 MM PIC 99.
        10 FILLER PIC X VALUE '-'.
        10 DD PIC 99.
    05 DEPARTMENT PIC AAA.
    05 MONTHLY-SALARY PIC 9(6).
```

(well-structure COBOL code for file read)

Applied occurrence of variable

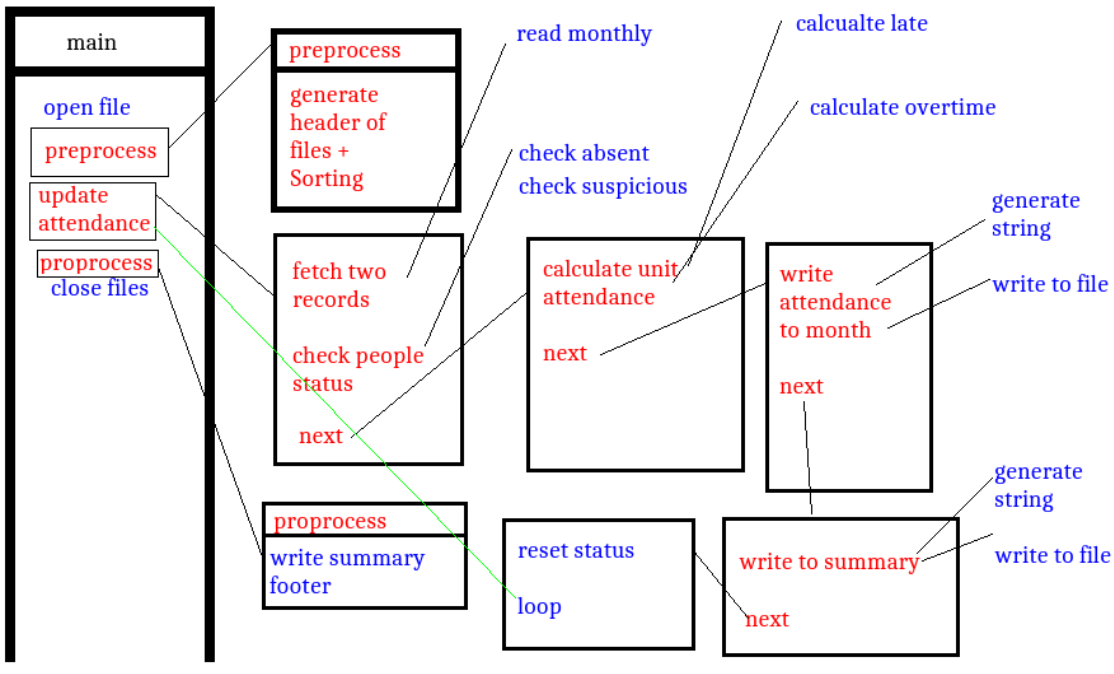
COBOL is much worse in variable referring. It would take long words in order to reference the variable if the variable name are set to be meaningful. Also the variable defined has to be found from the top section makes the readability decrease.
FORTRAN is better as it has no global variable, so that all variable must be defined in local scope. Although value of variable may change by other functions, the name stays the same. This facilitate reading the code section by section.

```
subroutine sortcond
  integer aRecTemp, aRecCount, aRec(10000,2)
  CHARACTER atxt*50
  common /coeff/ aRecTemp, aRecCount, aRec, atxt
  integer icount, loopcond, jcount
  common /sortloop/ icount, loopcond, jcount
  integer index
  index = jcount + 1
  if (aRec(index,1) .gt. aRec(index+1,1))
&  call swap(index,index+1)
 END
```

(unbalanced “Red” variable vs “Black” logic statement in FORTRAN)

2. Generally speaking, modern programming language like python is better than COBOL or FORTRAN since it has large community support. It has numerous resources(stack overflow Q&A) and packets to import in order to facilitate general tasks, while COBOL or FORTRAN syntex are only good for specialized tasks(file control/computation).
Also, scope of variable varies. Modern language basically supports both local and global variable, which COBOL only has global and FORTRAN only has local. This tiny modification of programming language hugely boosted the readability of the code.
Moreover, parameters passing and functions makes a grate deal. There are no recursion in FORTRAN and no parameters passing in COBOL. Which makes developer cannot use many programming techniques that are supported in python.

3. To separate tasks into sub-module, I utilizes the subroutine feature, trying to distinguish into “logic” sub-module and “utilite” sub-module. This helps me saves lots of time and save me from endless GOTO by using more CALL/PERFORM.



In the picture, both red and blue are subroutines. Red are ‘logic’ routine that only have CALL/PERFORM statement while blue are “utilities” subroutines which contains actual computation statement.