ADAM PROGRAMMING

- PSEUDO-CODE

1. Start
2. Open and read steps.in
3. If the file is not found, display an error message, if it exists, proceed to step 4.
4. Read the character on the first line as integer
5. Store the number as T
6. For number of T, read subsequent lines number of times
7. Create a variable to store number of steps
8. Read individual lines
9. Loop through every character of individual line
10. If the character is not "D", increment number of steps by 1
11. Display number of steps
12. End

- ALGORITHM

```
READ steps.in
IF steps.in is not found THEN
  DISPLAY file not found
ELSE
  READ first line of files.in
  SET value of first line TO T
  SET i to 0
  FOR i not greater than T
    READ a new line
    Initialize numOfSteps
    SET j to 0
    FOR j not greater than length of line
       IF character on line is not D THEN
            Increment numOfSteps
       ENDIF
    ENDFOR
  ENDIF
```

DUPLICATE FILES

- PSEUDO-CODE

13. Start
14. Open and read files.in
15. If the file is not found, display an error message, if it exists, proceed to step 4.
16. Read the character on the first line as integer
17. Store the number as T
18. For number of T, read subsequent lines number of times
19. Check if the next line is an integer and compute as group
20. Increment T until all groups are obtained
21. For a length of group, execute the following operation for the following lines in file.
22. For subsequent lines, split the name and ID
23. Store name and ID in current name and current variable.
24. Collect the name and ID of the next line
25. If the name of the current file and next file is not the same, add both current and next IDs to list of IDs
26. If current name is the same as next name, compare IDs
27. Store lowest ID
28. Display the sorted IDs
29. End

- ALGORITHM

```
READ files.in
IF files.in is not found THEN
  DISPLAY file not found
ELSE
  READ first line of files.in
  SET value of first line TO T
  SET i to 0
  FOR i not greater than T
    READ a new line
    IF new line is integer
      SET integer TO group
    ENDIF
    Initialize sortedIDs, lowestID
    SET j to 0

    NESTED-FOR j not greater than group
       Initialize currName, currID, nextName, nextID
       IF currName is  not the same as nextName
         Append currID and fileID to sortedIDs
       ELSE
         IF nextID is less than the currID
           SET lowestID to nextID
         ELSE
            SET lowestID to currID
       ENDIF
    END NESTED-FOR

    Sort fileIDs
    DISPLAY fileIDs
    DISPLAY lowestID
  ENDFOR
ENDIF
```

SELF DESCRIBING NUMBERS

- **PSEUDO-CODE**

1. Start
2. Open and read self.in

3. If the file is not found, display an error message, if it exists, proceed to step 4.
4. Read the character on the first line as integer
5. Store the number as T
6. For number of T, read subsequent lines number of times
7. Store line as with a variable, number
8. Obtain the length of number
9. For the length of number, loop through the character of number
10. Count the amount of occurrence for each character
11. Check if the index position is equal to the number of occurrence
12. If yes, display number is self describing
13. If no, display number is not self describing

- ALGORITHM

```
READ self.in
IF files.in is not found THEN
  DISPLAY file not found
ELSE
  READ first line of files.in
  SET value of first line TO T
  SET i to 0
  FOR i not greater than T
    READ a new line
    SET new line TO number

    NESTED-FOR each letter in number
       SET count TO 0
       SET numOfOcurrence TO letter at index position of number
       SET j to 0
       NESTED-FOR j not greater than length of number
          IF i is equal to the letter at j
             SET count TO count+1
          ENDIF
          IF count is equal to numOfOccurence for each number
```

```
            DISPLAY self describing
         ELSE
            DISPLAY not self describing
         ENDIF
      END NESTED-FOR
   ENDFOR
ENDIF
```