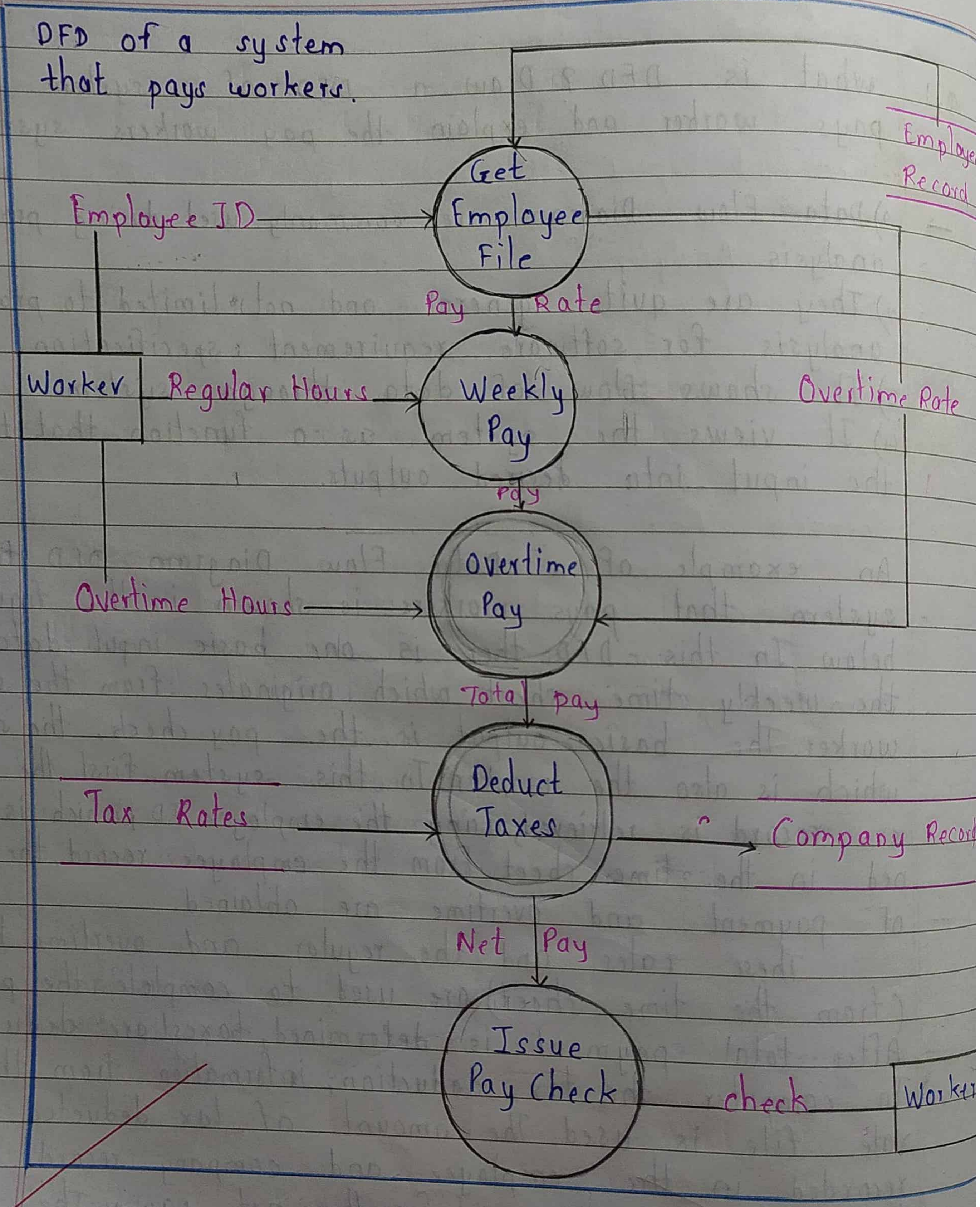
Assignment - 3

Q.1 What is DFD? Draw a DFD of a system that pays worker and explain the pay workers system?

→ 1) Data Flow Diagram are commonly used during problem analysis.
2) They are quite general and not limited to problem analysis for software requirement specification.
3) It shows flow of data through system.
4) It views the system as a function that transforms the input into desired outputs.

An example of a Data Flow Diagram - DFD for a system that pays workers is shown in the figure below. In this DFD there is one basic input data flow, the weekly time sheet, which originates from the source worker. The basic output is the pay check, the sink for which is also the worker. In this system, first the employees record is retrieved, using the employee ID, which is contained in the time sheet. From the employee record, the rate of payment and overtime are obtained.
These rates and the regular and overtime hours (from the time sheet) are used to complete the payment. After total payment is determined, taxes are deducted. To computer the tax deduction, information from the tax rate file is used. The amount of tax deducted is recorded in the employee and company records. Finally, the paycheck is issued for the net pay. The amount paid is also recorded in company records.

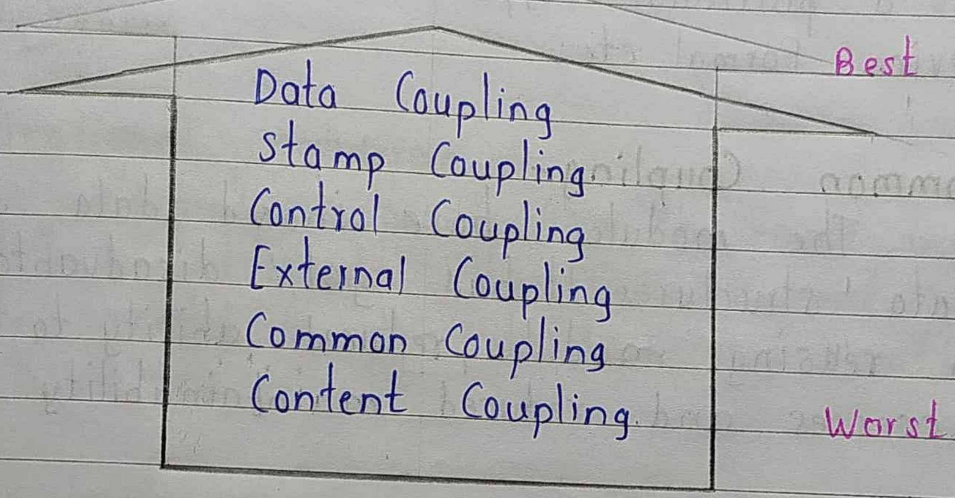


Q.2 What are different types of coupling and cohesion

→ • Coupling :-

Coupling is the measure of degree of

dependence between the modules. A good software will have low coupling.



- **Data Coupling :-**
  If the dependency between the modules is based on fact that they communicate by passing only data. then the modules are said to be data coupled. In data coupling, the components are independent of each other and communications don't contain tramp data. Eg. customer billing system.

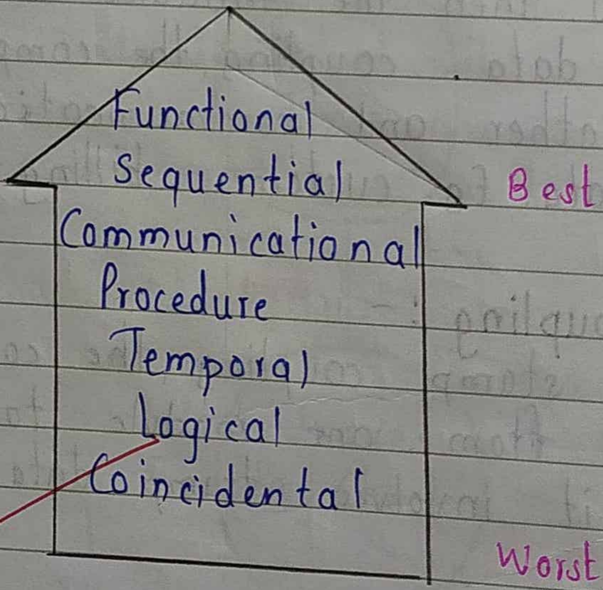
- **stamp coupling :-**
  In stamp coupling, the complete data structure is passed from one module to another module. Therefore, it involves tramp data.

- **Control Coupling :-**
  If the modules communicate by passing the control information. then they ~~have~~ are said to be controlled.
  ~~Eg. Protocol, External file, device format etc.~~
  Eg. Sort function that takes comparison function as an argument.

- External coupling -
  In external coupling, the modules depend on othe
modules, external to the software being developed or
to a particular type of hardware. E.g. Protocol, External Fi
device Formal etc.

- Common Coupling -
  The modules have shared data such as global
data structures. It has got disadvantages like difficu
in reusing modules, reduced ability to control data
accesses and reduced maintainability.

- Cohesion
  Cohesion is the measure of the degree to which
the elements of the module are functionally related
a good software design will have high cohesion.



- Functional Cohesion
  Elements are related to perform single function
A functional cohesion performs the tasks and
functions. It is an ideal situation.

classmate
Date
Page

• sequential cohesion-
An element outputs some data that becomes the input for other element i.e data flow between the parts. It occurs naturally in Functional Programming languages.

• communicational cohesion-
Two elements operate on the same input data or contribute towards the same output data
Eg. update record in the database and sent it to the printer.

• Procedural Cohesion-
Elements belong to common procedural unit. E.g. calculate student GPA. print student record, calculate cummatative GPA. print commulative GPA.

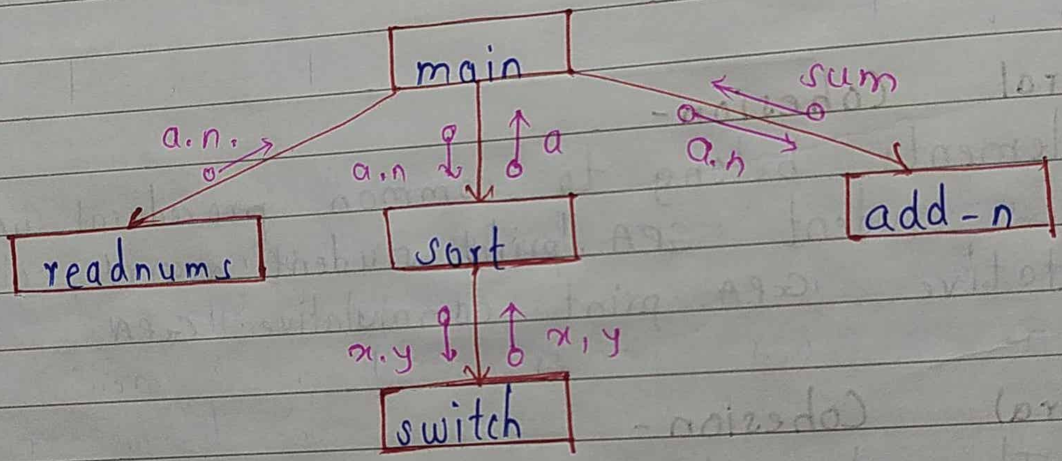
• Temporal Cohesion-
The elements are related by their timing involved. A module connected with temporal cohesion all the takes must be executed in the same time span This cohesion contains the code for initializing all the parts of the system. lots of different activities occur at the same time.

• Logical cohesion
~~The~~ elements are logically related e.g. A component reads input from the tape. disk and network. All the code for these functions. is in the some component operations are related but the functions are significance different

- coincidental cohesion -
  The element have no conceptabl relationship other than location in the source code. It is accide and worst form of cohesion. E.g. print next line an reverse the characters of a string in a single component.

Q.3 Draw a structured chart for sort program and sta open-closed principle with me example.

→



- open closed principle -

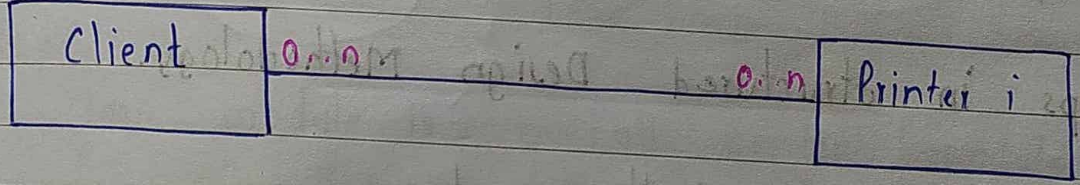
1) Besides cohesion and coupling, open-clased principle a helps in achieving modulating
2) Principle - ~~A~~ module should ~~bot~~ be open for extensio but closed for modification.
3) Behavior can be extended to accomodate requirem but existing code is not modified i.e it allows additi of code, but not modification of existing code.
4) Minimized risks of having existing functionality sto working due to changes - a very important considerat while changing code.
5) It is good for programmer as they like writing code.

c) In Object-oriented, this principle is satisfied by using inheritance and polymorphism.

v) Inheritance allows creating a new class to extend behavior without changing the original class. This can be used to support the open-closed principle.

Example -

Consider, an example of a client object which interacts with a printer object for printing



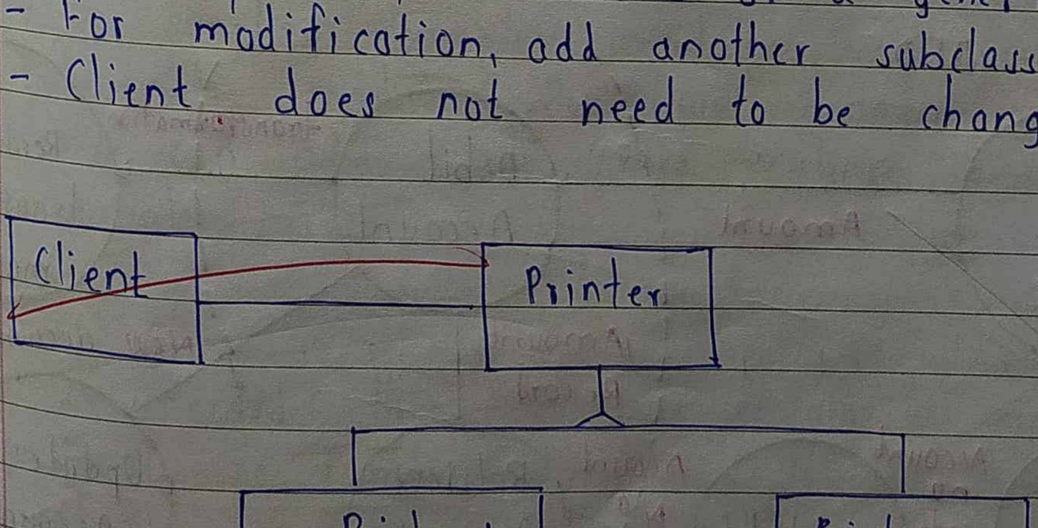
i) Client directly calls methods on printer1

ii) If another printer is to be allowed.

- a new class printer2 will be created.
- but the client will have to be changed if it wants to use Printer 2

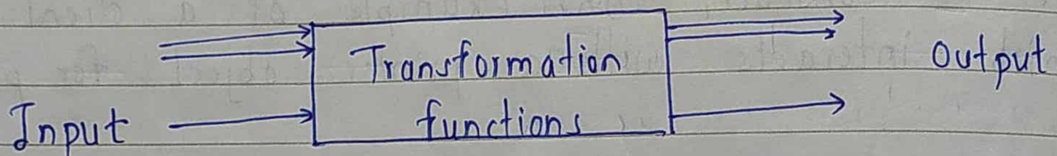
iii) Alternative approach -

- Have printer1, a subclass of a general printer.
- For modification, add another subclass, Printer 2.
- Client does not need to be changed.



Q.4 Give structured Design Methodology for ATM

→ • SDM –
SDM views software as a transformation functi
that converts given inputs to desire output
Goal – specify functional modules and connections object
-Low coupling and High Cohesion.

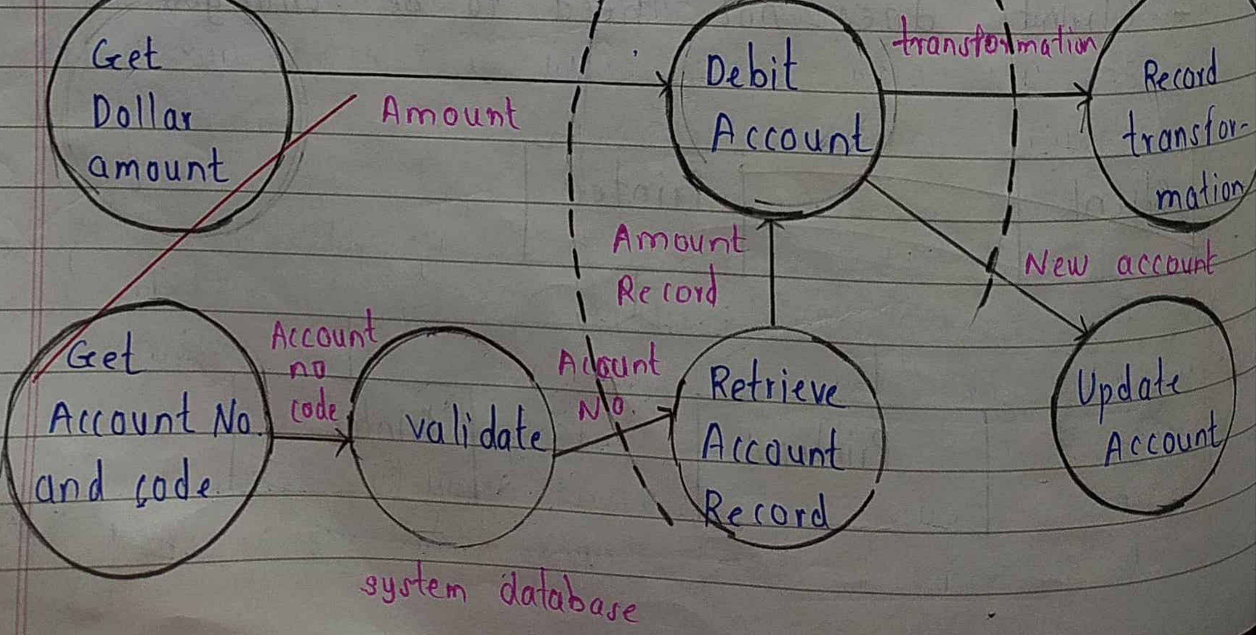


• Steps in structured Design Methodology.

1) Draw a DFD of the system.
2) Identify most abstract input and most abstract out
3) Fist level factoring
4) Factoring of input, output, transform modules.
5) Improving Design.

• SDM for an ATM Machine –

step 1 :- DFD



Date
Page

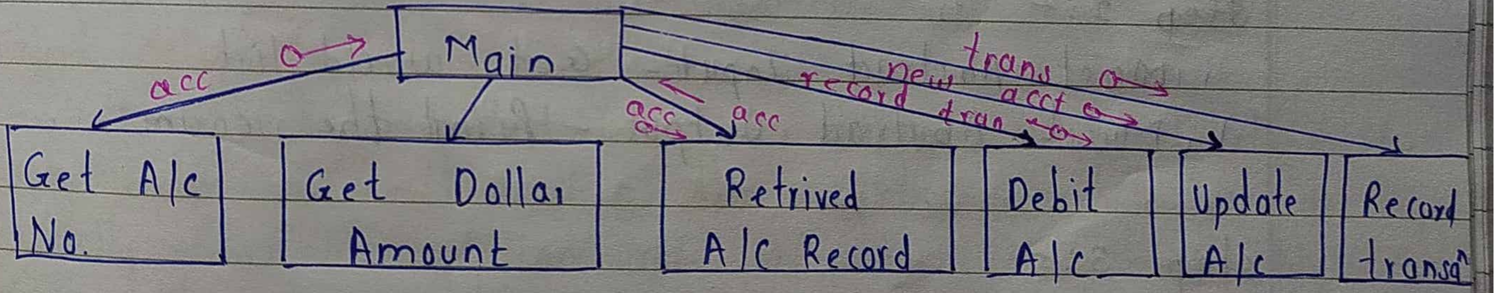
. step 2-

Two abstract inputs-
1. Get amount rupees
2. Validate account number

The validated account number is most abstract inpute than account number as it is still input.

Two abstract outputs-
1. Read the transaction
2. Update an account

. step. 3 -
Main module is overall control module.



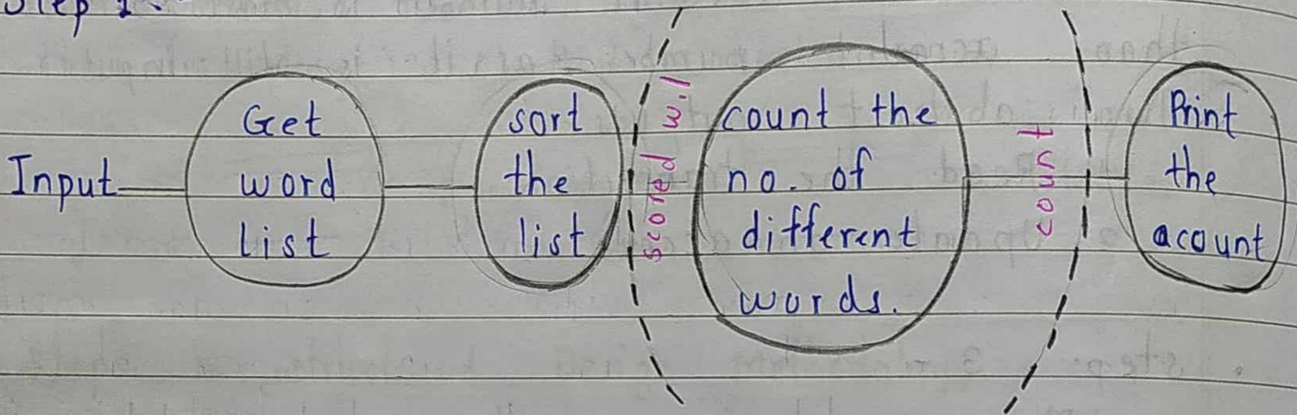
step 4:-
- To simplify complex modules they must be factored into subordinate modules that will distribute work of a module.
- we can add some extra features to simplify module if complex ~~ATM~~.

step 5:-
- If needed then structure should be modified.
- Goals of improving design is low coupling and high cohesion.
- Design heuristics used to modify initial design.
- A set of thumb rules that are generally usefull is

heuristics design

Q.5 Give SDM for problem statement " Determine the different number of words in an input file.

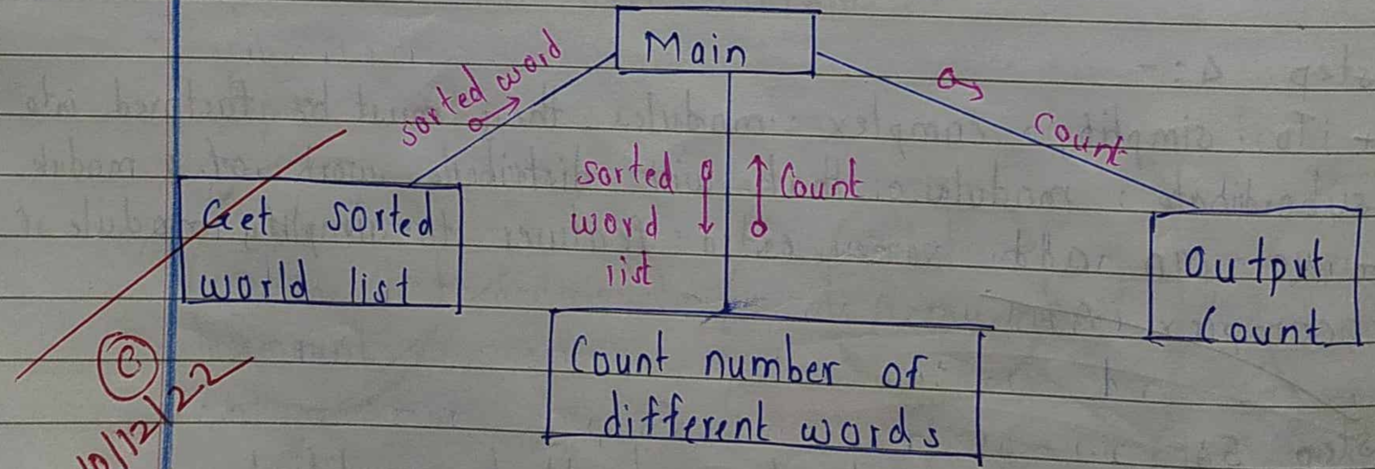
→ Step 1 :-



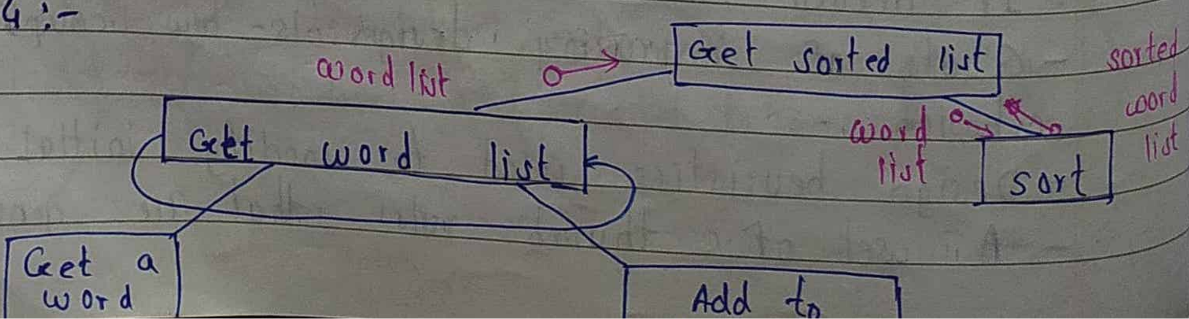
step 2 :-

Most Abstract Input - Get world list
Most Abstract Output - Print the count

step 3 :-



step 4 :-


Get sorted list
word list
sorted word list
word list
sort
Get word list
Get a word
Add to