Name of Experiment: You are going to develop a Chittagong University ERP solution based on three use cases: Recruiting Teachers and University Administrative staffs, enrolling students and publishing results. All the cases include Accounts, Exam Controller and Registrar Office. Accounts deal with the finance related works of students, teachers, officers and staffs. The recruitment of teachers is performed by viva. Officers recruitment will be conducted by both written exams and viva. And staffs are recruited by taking written exams, viva and practical exam. The enrollment of students will be performed by taking ~~Administrative~~ Admission test conducted by registrar office. Controller process results of students of 7 faculty: Science, Engineering, Life Science, Social Science, Arts, Business Administration and Law. Before publishing the result, Exam controller will check if the student has paid exam fees, Before checking the payment, accounts check if the student is valid.

Introduction:- We have to define three most important classes, Accounts, Exam Controller and Registrar office. Then we have to define a main class which will use the classes and demonstrate the usages.

Objectives:-

- how to use packages
- how to combine different classes to solve problems
- how to divide a problem into chunks.

Analysis : From analysing our problem we found below components of our problem.

- we have to define an Accounts class which checks if students are valid or not and checks if they have paid exam fees or not.
- a class ExamController, which takes help of an interface named Result, which publishes result of 7 faculties as per mentioned in the problem statement. Exam controller also publishes admission test result.
- a class, RegistnarOffice, that take help of an interface Recruitment, to recruit teachers, officers and staffs. It also takes help of a class student to take admission test.

From above analysis, the conceptual class diagrams are drawn in figure-1,

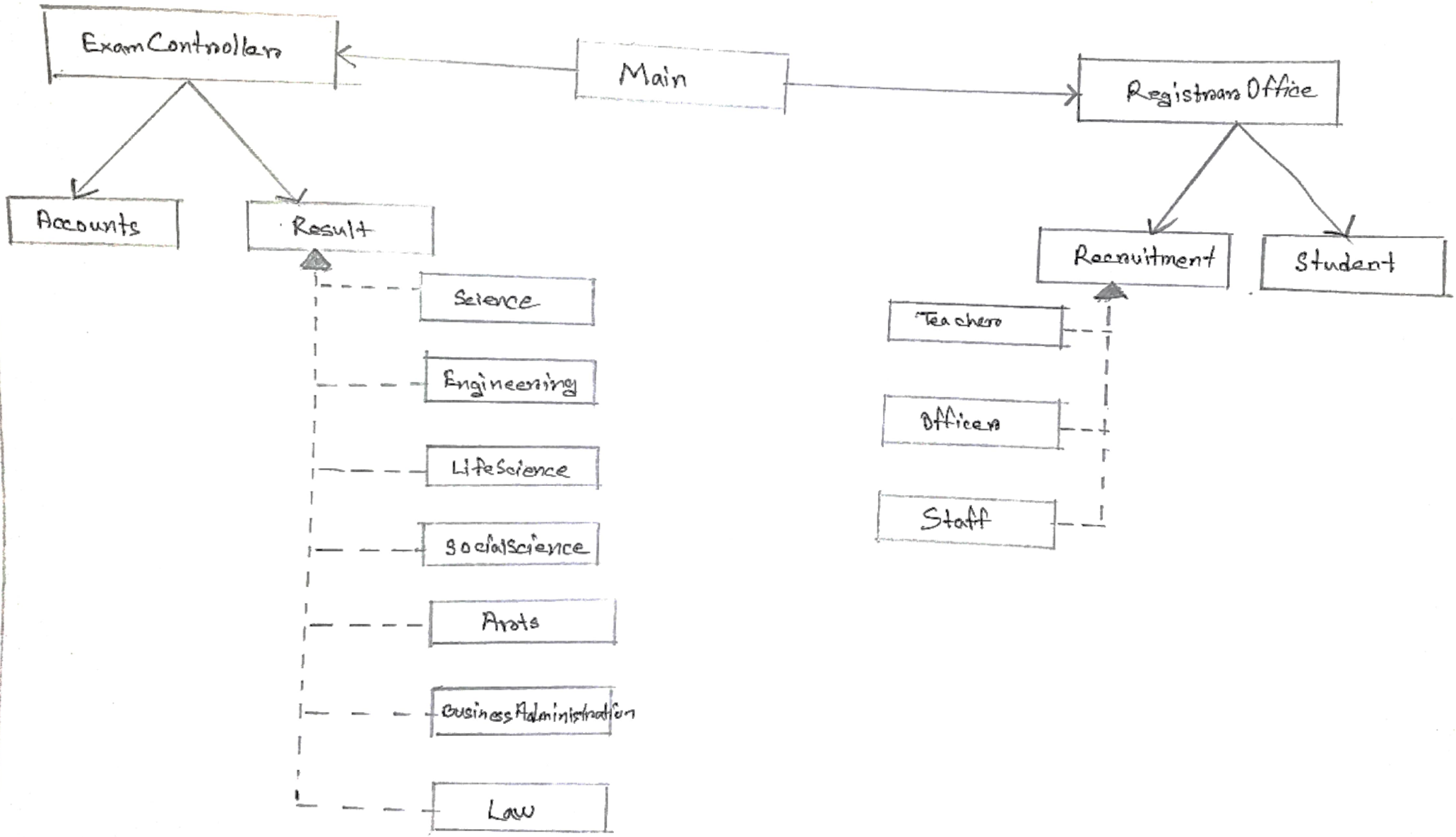

Figure-1 : conceptual class diagram

Design : From above analysis the ~~des~~ design descriptions are given below,

- A class Accounts : the class that handles all the finances of students
  - Methods :
    - areStudentsValid : checks if all the students are valid or not
    - haveStudents Paid Exam fees : checks if all the students have paid Exam fees.

- interface Result : the helper ~~class~~ interface that will help to publish results of different faculties.
  - Methods :
    - publishResult : different faculty ~~will~~ class will implement this method to publish their faculty result properly.

- class ExamController :
  - Data Members :
    - ac : an Account class object that will check for student validity
  - Methods :
    - processResults : processes and publishes different faculty results.
    - publishAddmissionTestResults : publishes addmission test results.

- class RegistrarOffice:
  - interface Recruitment: helper interface to recruit teacher, staff, officer and enroll students.
    - Methods:
      - recruit: the derived classes will implement this method

- class Teacher:
  - Methods:
    - recruit: recruitment process for teachers
    - viva: viva for teacher recruitment.

- class Officer:
  - Methods:
    - recruit: recruitment process for officers
    - writtenExam: written exam for officer recruitment
    - viva: viva for officer recruitment

- class Staff:
  - Methods:
    - recruit: recruitment process for staffs
    - writtenExam: written exam for staffs recruitment
    - practicalExam: practical exam for staffs recruitment
    - viva: viva for staffs recruitment

- class Student:
  - Methods:
    - takeAdmissionTest: takes students admission tests.

- class RegistrarOffice:
  - Data Members:
    - r : a Recruitment type object, that will recruit objects as per request.
  - Methods:
    - recruitTeacher : recruitment for teacher
    - recruitOfficer : recruitment for officer
    - recruitStaff : recruitment for staff
    - ~~recruitSt~~ enrollStudent : enrollment for student.

From above design description, the pseudocodes for methods are given below in Figure-2

architectural class design is

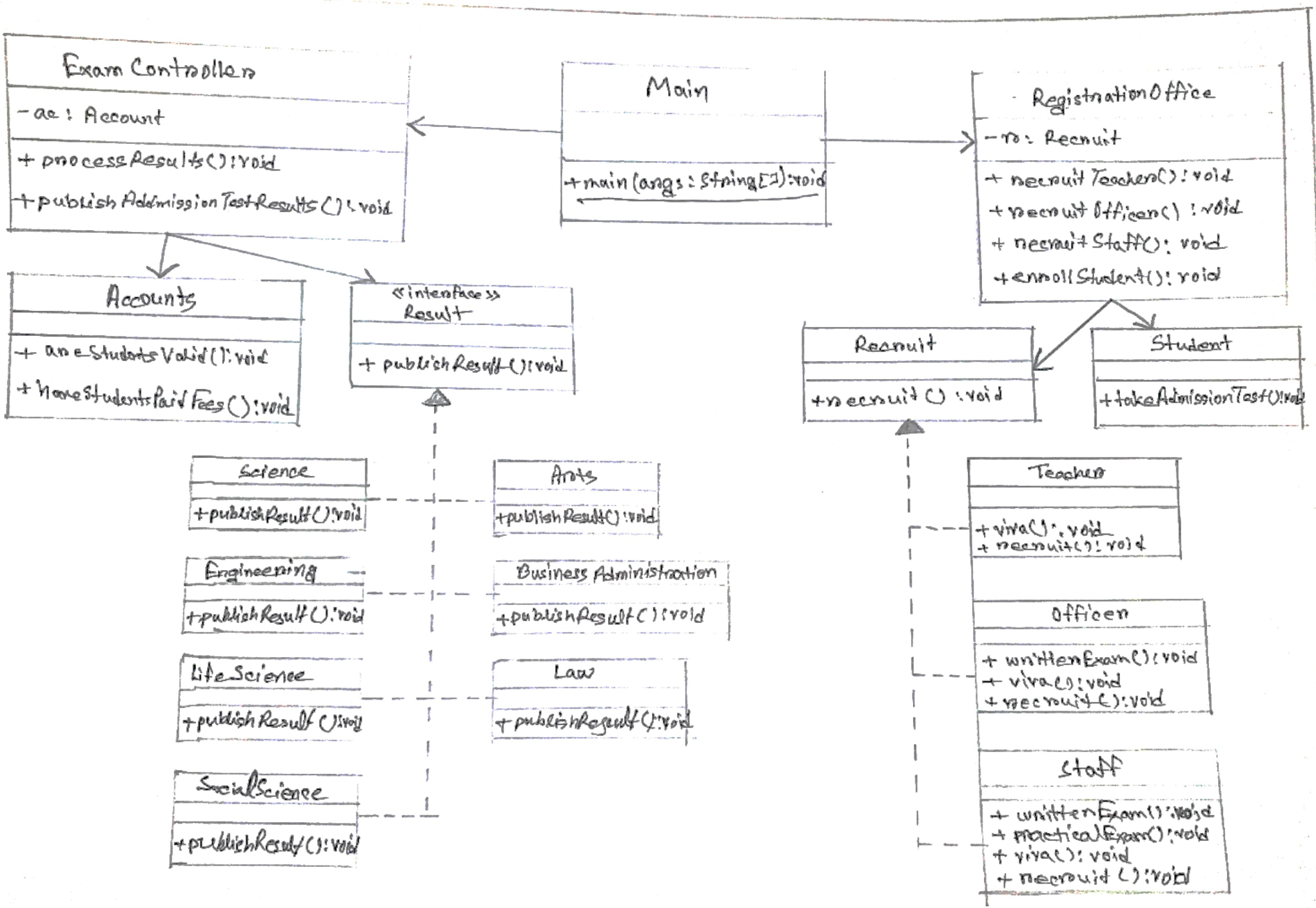

Figure:2: architectural class design

From above design description the pseudo codes of methods are given below.

processResults():
assign all faculty class object to a variable of type Result, then publish it

publishAdmissionTestResult():
print the admission result published message

areStudentsValid():
check if all students are valid or not

haveStudentPaidFees():
check if students are valid or not, then check if they have paid fees or not

publishResult():
individual classes prints the name of the faculty and prints the result published message

recruit():
officer, Teache, and Staff class implement their own way of recruiting

recruitTeacher():
calls Teacher classes recruit method

recruitOfficer():
calls Officer class's recruit method

recruitStaff():
calls Staff class's recruit method

enrollStudent():
calls Student class's. takeAddmissionTest method

takeAddmissionTest():
prints the addmission test taken message.

Implementation:
* implementation is attached with the report *

Conclusion:
we defined three important classes of the problem and some helper interfaces and classes. Then we defined the Main class and main method inside it and used all the classes to demonstrate the solution.