TITLE OF THE PROJECT

Submitted by

Name of the Students: Aritra Ghosal
Enrolment number: 12022002018036
Section: F
Class Roll Number: 28
Stream: C.S.B.S
Subject: Programming for Problem Solving
Subject Code: IVC101
Department: Basic Science and Humanities

Under the supervision of
Name of the teachers

Academic Year: 2022-26

PROJECT REPORT SUBMITTED IN PARTIAL FULFILLMENT OF THE

REQUIREMENTS FOR THE FIRST SEMESTER
DEPARTMENT OF BASIC SCIENCE AND HUMANITIES
INSTITUTE OF ENGINEERING AND MANAGEMENT, KOLKATA

CERTIFICATE OF RECOMMENDATION

We hereby recommed that the project prepared under our supervision by Aritra Ghosal, entitled Title of the Project be accepted in partial fulfillment of the requirements for the degree of partial fulfillment of the first semester.

Head of the Department Basic Sciences and Humanities IEM, Kolkata	Project Supervisor

1 Introduction

Python is a versatile and easy to use language often used in data manipulation. What separates Python from all other languages is its large number of use cases. Whereas Javascript is used for the web, C for systems, R for data, Python can be used for all three and many more. The following project demonstrates a model system run using mainly python.

1.1 Objective

This project attempts to model a small scale database management system utilized by an academic institution. The objective of this project is to learn and demonstrate several python programming concepts including:

- Using python code from other files
- Importing and using third party modules
- Reading and writing text files
- Managing CSV data
- Plotting data
- Building a basic user interface
- Utilizing concepts of Object Oriented Programming

This project also demonstrates general programming concepts such as ER diagrams.

1.2 Organization of the Project

```
.
├── code
│   ├── batch.py
│   ├── course.py
│   ├── department.py
│   ├── examination.py
│   ├── main.py
│   └── student.py
├── databases
│   ├── batch.csv
│   ├── course.csv
│   ├── department.csv
│   └── student.csv
├── fonts
│   ├── CONSOLAB.TTF
│   ├── timesnewroman-bold.ttf
│   ├── timesnewroman-bolditalic.ttf
│   ├── timesnewroman-italic.ttf
│   └── timesnewroman-regular.ttf
├── instructions
│   ├── IEM logo.png
│   ├── Project Details.docx
│   └── Project Report Template.docx
├── latex
│   ├── er-diagram.sty
│   ├── er.tex
│   ├── project.tex
│   └── template.tex
├── Makefile
└── outputs
    ├── output.log
    └── …
```

The **code** directory contains all the python code that is being executed at runtime. **batch.py** is a module that exports functions that operate on a batch. Likewise, **course.py** is a module that exports functions that operate on courses in the database. Same for **department.py**, which is a module that exports functions that operate on a department. **examination.py** exports the **Examination** class that represents an examination being held by the institution. **main.py** is a file with executive permissions which imports all of the above and runs a simple menu based command line user interface.

The **databases** directory contains all the data in CSV format.

The **fonts** directory contains the fonts required to compile this document.

The **instructions** directory contains all of the raw material to given to build this project.

The **latex** directory contains all of the LaTeX code used to build the project report (this

file). **template.tex** sets the default values necessary for the project report. **project.tex** contains the code that is compiled into the project report. It contains sources the outputs and diagrams along with the python code to include in the project report.
er.tex contains the er diagram for the database and **er-diagram.sty** is a third party library used to draw the er diagram.

The **Makefile** contains the build system for the entire project. It specifies the dependencies for each component and runs the commands to create each component. The **Makefile** also contains code that generates the databases and fills them with random data modelling the system as closely as possible. This is the centre point of the entire project, it determines the order and execution of everything else in the project.

The **outputs** directory contains all of the output generated by the python code at runtime. The **output.log** file is generated file running the python code, it contains the entire interaction between the program and the user via the command line interface and stores it for future reference.

2 Database Descriptions

Each student in the **student.csv** database has a unique ID, along with a name and a class roll number. Each student is associated with a single batch.
Each batch in **batch.csv** is assigned a unique ID. They also have name and a department they fall under. Each batch has a list of courses and a list of students who appear for the courses.
Each course in **course.csv** has an ID, subject name and a storage of marks obtained by each student appearing for the course.
Each department in **department.csv** has an ID, name and list of batches that worked under that department.

2.1 Database Samples

batch.csv

Batch ID	Batch Name	Department Name	List of Courses	List of Students
CSE00	CSE 2000-2004	CSE	…	…
CSE01	CSE 2001-2005	CSE	…	…
CSE02	CSE 2002-2006	CSE	…	…
CSE04	CSE 2004-2008	CSE	…	…
CSE05	CSE 2005-2009	CSE	…	…
CSE08	CSE 2008-2012	CSE	…	…
CSE09	CSE 2009-2013	CSE	…	…

CSE13	CSE 2013-2017	CSE	…	…
CSE14	CSE 2014-2018	CSE	…	…
CSE21	CSE 2021-2025	CSE	…	…
CSE91	CSE 1991-1995	CSE	…	…
CSE92	CSE 1992-1996	CSE	…	…
CSE93	CSE 1993-1997	CSE	…	…
CSE94	CSE 1994-1998	CSE	…	…
CSE95	CSE 1995-1999	CSE	…	…
CSE96	CSE 1996-2000	CSE	…	…
CSE97	CSE 1997-2001	CSE	…	…
CSE98	CSE 1998-2002	CSE	…	…
ECE02	ECE 2002-2006	ECE	…	…
ECE05	ECE 2005-2009	ECE	…	…
ECE06	ECE 2006-2010	ECE	…	…
ECE10	ECE 2010-2014	ECE	…	…
ECE11	ECE 2011-2015	ECE	…	…
ECE13	ECE 2013-2017	ECE	…	…
ECE15	ECE 2015-2019	ECE	…	…
ECE19	ECE 2019-2023	ECE	…	…
ECE90	ECE 1990-1994	ECE	…	…
ECE97	ECE 1997-2001	ECE	…	…
ECE98	ECE 1998-2002	ECE	…	…
ECE99	ECE 1999-2003	ECE	…	…
IT02	IT 2002-2006	IT	…	…
IT04	IT 2004-2008	IT	…	…
IT09	IT 2009-2013	IT	…	…
IT12	IT 2012-2016	IT	…	…
IT13	IT 2013-2017	IT	…	…
IT14	IT 2014-2018	IT	…	…
IT15	IT 2015-2019	IT	…	…
IT16	IT 2016-2020	IT	…	…
IT17	IT 2017-2021	IT	…	…
IT18	IT 2018-2022	IT	…	…
IT90	IT 1990-1994	IT	…	…
IT93	IT 1993-1997	IT	…	…
IT96	IT 1996-2000	IT	…	…
IT98	IT 1998-2002	IT	…	…

IT22	IT 2022-2026	IT	…	…

course.csv

Course ID	Course Name	Marks Obtained
CSE00	CSE 2000-2004	CSE
CSE01	CSE 2001-2005	CSE
CSE02	CSE 2002-2006	CSE
CSE04	CSE 2004-2008	CSE
CSE05	CSE 2005-2009	CSE
CSE08	CSE 2008-2012	CSE
CSE09	CSE 2009-2013	CSE
CSE13	CSE 2013-2017	CSE
CSE14	CSE 2014-2018	CSE
CSE21	CSE 2021-2025	CSE
CSE91	CSE 1991-1995	CSE
CSE92	CSE 1992-1996	CSE
CSE93	CSE 1993-1997	CSE
CSE94	CSE 1994-1998	CSE
CSE95	CSE 1995-1999	CSE
CSE96	CSE 1996-2000	CSE
CSE97	CSE 1997-2001	CSE
CSE98	CSE 1998-2002	CSE
ECE02	ECE 2002-2006	ECE
ECE05	ECE 2005-2009	ECE
ECE06	ECE 2006-2010	ECE
ECE10	ECE 2010-2014	ECE
ECE11	ECE 2011-2015	ECE
ECE13	ECE 2013-2017	ECE
ECE15	ECE 2015-2019	ECE
ECE19	ECE 2019-2023	ECE
ECE90	ECE 1990-1994	ECE
ECE97	ECE 1997-2001	ECE
ECE98	ECE 1998-2002	ECE
ECE99	ECE 1999-2003	ECE
IT02	IT 2002-2006	IT

IT04	IT 2004-2008	IT
IT09	IT 2009-2013	IT
IT12	IT 2012-2016	IT
IT13	IT 2013-2017	IT
IT14	IT 2014-2018	IT
IT15	IT 2015-2019	IT
IT16	IT 2016-2020	IT
IT17	IT 2017-2021	IT
IT18	IT 2018-2022	IT
IT90	IT 1990-1994	IT
IT93	IT 1993-1997	IT
IT96	IT 1996-2000	IT
IT98	IT 1998-2002	IT
IT22	IT 2022-2026	IT

department.csv

Department ID	Department Name	List of Batches
CSE	Computer Science and Engineering	…
ECE	Electronics and Communication Engineering	…
IT	Information Technology	…
BA	Business Administration	…

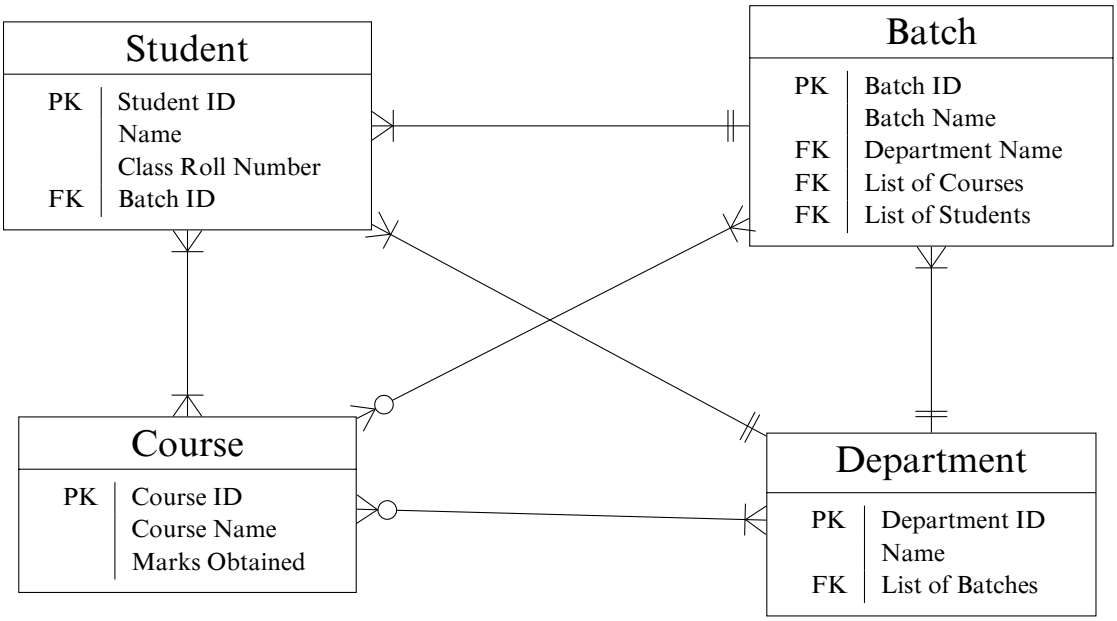
student.csv

Student ID	Name	Class Roll No	Batch ID
CSE0526	Rati Koshy	E-95	CSE05
ECE9097	Manjari Aggarwal	E-39	ECE90
IT1373	Zara Badal	B-69	IT13
ECE1139	Anvi Dash	G-41	ECE11
ECE1535	Jayant Joshi	D-87	ECE15
IT9326	Mahika Golla	G-25	IT93
ECE1534	Ayesha Kunda	C-45	ECE15
CSE9412	Mannat Loke	H-33	CSE94
ECE1985	Miraan Mandal	F-57	ECE19

CSE9844	Vivaan Bhatti	A-42	CSE98
CSE0983	Zara Vig	A-34	CSE09
CSE9765	Aarush Randhawa	E-18	CSE97
ECE1551	Parinaaz Choudhary	B-09	ECE15
ECE0235	Jivika Master	F-56	ECE02
CSE0818	Vardaniya Walia	G-61	CSE08
IT1206	Neysa Tandon	B-61	IT12
CSE0234	Tara Garde	F-54	CSE02
CSE9653	Lakshit Thakur	A-43	CSE96
IT1529	Hiran Mandal	A-82	IT15
ECE9754	Gatik Sama	B-08	ECE97
CSE0466	Ira Chandran	C-94	CSE04
CSE1494	Jayesh Walia	B-61	CSE14
ECE1009	Kabir Hans	E-42	ECE10
CSE2111	Indrans Kannan	A-83	CSE21
CSE9285	Misha Ghose	G-66	CSE92
ECE0559	Nehmat Rattan	D-64	ECE05
CSE9109	Shamik Sibal	A-80	CSE91
CSE1342	Hiran Manda	G-85	CSE13
ECE1029	Kanav Sagar	A-43	ECE10
IT1770	Prerak Raju	F-62	IT17
IT9850	Hazel Gala	D-74	IT98
IT9058	Jivin Varty	F-84	IT90
ECE0698	Yashvi Jhaveri	C-12	ECE06
IT1231	Yashvi Acharya	A-20	IT12
IT1845	Siya Hegde	F-18	IT18
ECE1350	Kismat Dora	D-12	ECE13
ECE9840	Aniruddh Ratti	F-87	ECE98
IT1378	Stuvan Devi	H-95	IT13
IT9659	Priyansh Walia	C-29	IT96
CSE0206	Mannat Gara	A-35	CSE02
CSE9280	Onkar Baral	C-30	CSE92
CSE9541	Mohanlal Hayre	G-33	CSE95
IT0926	Faiyaz Tella	G-08	IT09
ECE9983	Amira Malhotra	D-39	ECE99
CSE0295	Mehul Sem	C-82	CSE02
CSE0175	Saksham Khosla	F-92	CSE01

IT1400	Samarth Agarwal	C-34	IT14
ECE1592	Shaan Chawla	F-18	ECE15
IT9894	Hridaan Bains	C-58	IT98
IT0439	Gatik Hayer	E-00	IT04
IT0283	Anya Chawla	C-63	IT02
IT1611	Shlok Bir	C-71	IT16
CSE9308	Myra Bir	F-20	CSE93
CSE0080	Yashvi Khurana	D-56	CSE00
IT9624	Kartik Joshi	B-22	IT96

3 E-R Diagram



4 Programs

main.py

```
#!/bin/python3
from re import search
#import from modules
from student import
 ↪  create_student,update_student,remove_student,report
from course import create_course,course_performance,course_statistics
from batch import
 ↪  create_batch,students,courses,batch_performance,batch_statistics
```

```
from department import
↪  create_department,batches,batch_averages,department_statistics
from examination import Examination
def input_marks():
    while True:
        roll_number=input('\n\t\t\tClass Roll Number: ')
        if roll_number=='':
            break
        yield {
            'roll number':roll_number,
            'name':input('\t\t\tStudent Name: '),
            'marks':float(input('\t\t\tMarks: '))
        }
def input_array(data,id):
    print(f'\t\t\tEnter the {data} for {id}')
    while True:
        data=input('\t\t\t\t: ')
        if data=='':break
        yield data
while True:
    choice=input('''
1. Student
2. Course
3. Batch
4. Department
5. Examination
: ''')
    if choice=='':break
    elif choice=='1':
        choice=input('''
    1. Create a new student
    2. Update details of a student
    3. Remove a student
    4. Generate report of a student
    : ''')
        if choice=='1':
            create_student(
                student_id=input('\t\tStudent ID: '),
                name=input('\t\tStudent Name: '),
                class_roll_no=input('\t\tClass Roll No: '),
                batch=input('\t\tBatch ID: ')
            )
        elif choice=='2':
            update_student(
                student_id=input('\t\tStudent ID: '),
                name=input('\t\tStudent Name: '),
                class_roll_no=input('\t\tClass Roll No: '),
            )
        elif choice=='3':
            remove_student(
                student_id=input('\t\tStudent ID: ')
            )
        elif choice=='4':
            report(
                student_id=input('\t\tStudent ID: ')
            )
    elif choice=='2':
        choice=input('''
    1. Create a new course
    2. View performance of all students
```

```
3. Create course statistics
:''')
    if choice=='1':
        create_course(
            course_id=input('\t\tCourse ID: '),
            course_name=input('\t\tCourse Name: '),
            marks=[student for student in input_marks()]
        )
    elif choice=='2':
        course=input('\t\tCourse: ')
        if search('^C0[0-9]{2}$',course):
            for i in course_performance(course_id=course):
                print('\t\t\t',i)
        else:
            for i in course_performance(course_name=course):
                print('\t\t\t',i)
    elif choice=='3':
        course=input('\t\tCourse: ')
        if search('^C0[0-9]{2}$',course):
            course_statistics(course_id=course)
        else:
            course_statistics(course_name=course)
elif choice=='3':
    choice=input('''
1. Create a new batch
2. View list of students in a batch
3. View list of courses taught in a batch
4. View performance of a batch
5. Create pie chart of percentage of all students
: ''')
    batch_id=input('\t\tBatch ID: ')
    if choice=='1':
        create_batch(
            batch_id=batch_id,
            batch_name=input('\t\tBatch Name: '),
            department_name=input('\t\tDepartment Name: '),
            courses=[i for i in input_array('courses',batch_id)],
            students=[i for i in input_array('students',batch_id)]
        )
    elif choice=='2':
        print('\t\t',students(batch_id=batch_id))
    elif choice=='3':
        print('\t\t\t',courses(batch_id=batch_id))
    elif choice=='4':
        for i in
        ↪ batch_performance(batch_id=batch_id):print('\t\t\t',i)
    elif choice=='5':
        batch_statistics(batch_id=batch_id)
elif choice=='4':
    choice=input('''
1. Create a new department
2. View batches of a department
3. View average performance of batches of a department
4. Create statistics of a department
:''')
    department_id=input('\t\tDepartment ID: ')
    if choice=='1':
        create_department(
            department_id=department_id,
```

```
                department_name=input('\t\tDepartment Name: '),
                batches=[i for i in
                 ↪  input_array('batches',department_id)]
            )
        elif choice=='2':
            print('\t\t\t',batches(department_id=department_id))
        elif choice=='3':
            for i in batch_averages(department_id=department_id):
                print(i)
        elif choice=='4':
            department_statistics(department_id=department_id)
    elif choice=='5':
        print('''
    Hold an examination:
''')
        exam=Examination(*[i for i in input_array('batches','exam')])
        choice=input('''
    1. View student perfomance in the examination
    2. Create examination statistics
    : ''')
        if choice=='1':
            print(exam.student_performance)
        elif choice=='2':
            exam.statistics()
```

---

student.py

```
from csv import writer,reader
from texttable import Texttable
def create_student(**kwargs):
    batch_id=kwargs['batch']
    student_id=kwargs['student_id']
    with open('databases/student.csv','a') as csvfile:
        writer(csvfile).writerow([
            student_id,
            kwargs['name'],
            kwargs['class_roll_no'],
            batch_id
        ])
    rows=[]
    with open('databases/batch.csv','r') as csvfile:
        for row in reader(csvfile):
            if row[0]==batch_id:
                row[4]+=f':{student_id}'
            rows.append(row)
    with open('databases/batch.csv','w') as csvfile:
        db=writer(csvfile)
        for row in rows:
            db.writerow(row)
def update_student(**kwargs):#update by student id
    rows=[]
    EXIT_CODE=1
    with open('databases/student.csv','r') as csvfile:
        db=reader(csvfile)
        for row in db:
            if row[0]==kwargs['student_id']:
```

```
                EXIT_CODE=0
                rows.append([
                    row[0],
                    kwargs['name'] if 'name' in kwargs else row[1],
                    kwargs['class_roll_no'] if 'class_roll_no' in
                     ↪  kwargs else row[2],
                    kwargs['student_id'][:-2]
                ])
                break
            rows.append(row)
        for row in db:rows.append(row)#add remaining
    with open('databases/student.csv','w') as csvfile:#update file
        db=writer(csvfile)
        for row in rows:db.writerow(row)
    return EXIT_CODE
def remove_student(student_id):#remove by student id
    rows=[]
    EXIT_CODE=1
    with open('databases/student.csv','r') as csvfile:
        db=reader(csvfile)
        for row in db:
            if row[0]==student_id:#found
                batch_id=row[3]
                EXIT_CODE=0
                break
            rows.append(row)
        for row in db:rows.append(row)#add remaining
    with open('databases/student.csv','w') as csvfile:#update file
        db=writer(csvfile)
        for row in rows:db.writerow(row)
    if EXIT_CODE==1:return 1#student not found
    rows=[]
    empty_batch=False
    with open('databases/batch.csv','r') as csvfile:
        db=reader(csvfile)
        for row in db:
            if row[0]==batch_id:
                students=row[4].split(':')
                students.remove(student_id)
                courses=row[3].split(':')
                if len(students)==0:
                    empty_batch=True
                    department_name=row[2]
                else:
                    row[4]=':'.join(students)
                    rows.append(row)
                break
            rows.append(row)
        for row in db:rows.append(row)
    with open('databases/batch.csv','w') as csvfile:
        db=writer(csvfile)
        for row in rows:db.writerow(row)
    rows=[]
    with open('databases/course.csv','r') as csvfile:
        db=reader(csvfile)
        for row in db:
            if row[0] in courses:
```

```
                marks=row[2]
                a=marks.index(student_id)
                b=marks.find('-',a)
                row[2]=marks[:a-1]+marks[b:]
            rows.append(row)
    with open('databases/course.csv','w') as csvfile:
        db=writer(csvfile)
        for row in rows:db.writerow(row)
    if not empty_batch:return 0
    rows=[]
    with open('databases/department.csv','r') as csvfile:
        db=reader(csvfile)
        for row in db:
            if row[0]==department_name:
                batches=row[2].split(':')
                batches.remove(batch_id)
                row[2]=':'.join(batches)
                rows.append(row)
                break
            rows.append(row)
        for row in db:rows.append(row)
    with open('databases/department.csv','w') as csvfile:
        db=writer(csvfile)
        for row in rows:db.writerow(row)
def report(student_id):
    def grade(marks):
        if marks>=90:grade='A'
        elif marks>=80:grade='B'
        elif marks>=70:grade='C'
        elif marks>=60:grade='D'
        elif marks>=50:grade='E'
        else: return 'F','Failed'
        return (grade,'Passed')
    EXIT_CODE=1
    with open('databases/student.csv') as csvfile:
        db=reader(csvfile)
        for row in db:
            if row[0]==student_id:
                _,name,roll,batch_id=row
                EXIT_CODE=0
                break
    if EXIT_CODE==1:return 1
    with open('databases/batch.csv') as csvfile:
        db=reader(csvfile)
        for row in db:
            if row[0]==batch_id:
                exams=row[3].split(':')
                break
    marksheet=Texttable()
    marksheet.set_cols_align(('l','l','r','r','c','l'))
    marksheet.add_row(['Course','Course Id','Marks Obtained','Full
    ↪  Marks','Grade','Remarks'])
    total=0
    with open('databases/course.csv') as csvfile:
        db=reader(csvfile)
        for row in db:
            if row[0] in exams:
```

```
                performance=row[2]
                i=performance.index(student_id)
                a=performance.find(':',i)
                b=performance.find('-',i)
                marks=float(performance[a+1:b])
                total+=marks
                marksheet.add_row([
                    row[1],
                    row[0],
                    marks,
                    100,
                    *grade(marks)
                ])
    number=len(exams)
    marksheet.add_row(['Total','-',total,number*100,*grade(total/numb
     ↪  er)])
    with open(f'outputs/{student_id}-report_card.txt','w') as
     ↪  report:report.write(f'''
{name} ({roll})
{marksheet.draw()}
ID:{student_id}
Batch:{batch_id}
 ''')
    return EXIT_CODE
```

course.py

```
from csv import reader,writer
from collections import namedtuple
from matplotlib.pyplot import
 ↪  hist,title,xlabel,ylabel,xticks,xlim,style,close,savefig
def _parse_args(argdict):
    wrong_arg=Exception('Either provide course_id or course_name')
    if len(argdict)>1:raise wrong_arg
    (param,val),=argdict.items()
    if param=='course_id':rown=0
    elif param=='course_name':rown=1
    else:raise wrong_arg
    return rown,val
def create_course(**kwargs):
    marks=''
    batches=set()
    course_id=kwargs['course_id']
    with open('databases/student.csv','r') as csvfile:
        db=reader(csvfile)
        for student_data in kwargs['marks']:
            roll=student_data['roll number']
            for row in db:
                if row[2]==roll:
                    student_id=row[0]
                    marks+=f"{student_id}:{student_data['marks']}-"
                    batches.add(student_id[0:-2])
                    csvfile.seek(0)
                    break
    with open('databases/course.csv','a') as csvfile:
```

```
        writer(csvfile).writerow([
            course_id,
            kwargs['course_name'],
            marks[:-1]#skip last '-'
        ])
    rows=[]
    with open('databases/batch.csv','r') as csvfile:
        for row in reader(csvfile):
            if row[0] in batches:
                row[3]+=':'+course_id
            rows.append(row)
    with open('databases/batch.csv','w') as csvfile:
        db=writer(csvfile)
        for row in rows:db.writerow(row)
def course_performance(**kwargs):
    rown,val=_parse_args(kwargs)
    Student=namedtuple("Student",('roll','name','marks'))
    marks=False
    with open('databases/course.csv','r') as csvfile:
        for row in reader(csvfile):
            if row[rown]==val:
                marks=row[2].split('-')
                break
    if not marks:return -1
    with open('databases/student.csv','r') as csvfile:
        db=reader(csvfile)
        for perf in marks:
            a=perf.index(':')
            student_id=perf[:a]
            for row in db:
                if row[0]==student_id:
                    yield Student(row[2],row[1],float(perf[a+1:]))
                    csvfile.seek(0)#start from beginning
                    break
def course_statistics(**kwargs):
    rown,val=_parse_args(kwargs)
    marks=False
    with open('databases/course.csv','r') as csvfile:
        for row in reader(csvfile):
            if row[rown]==val:
                performance=row[2]
                if performance=='':return -1
                marks=[float(i[i.index(':')+1:]) for i in
                 ↪  performance.split('-')]
                break
    if not marks:return -1
    style.use('Solarize_Light2')
    hist(marks,bins=[0,50,60,70,80,90,100])
    title(val)
    xlabel('marks')
    ylabel('number of students')
    xticks([25,55,65,75,85,95],['F','E','D','C','B','A'])
    xlim(100,0)
    savefig(f'outputs/Course Statistics-{val}.png')
    close()
```

```
from csv import reader,writer
from functools import partial
from collections import namedtuple
from matplotlib.pyplot import
 ↪  pie,title,style,xticks,yticks,close,savefig
Student=namedtuple("Student",('roll','name','percentage'))
def _parse_args(argdict):
    wrong_arg=Exception('Either provide batch_id or batch_name')
    if len(argdict)>1:raise wrong_arg
    (param,val),=argdict.items()
    if param=='batch_id':rown=0
    elif param=='batch_name':rown=1
    else:raise wrong_arg
    return rown,val
def _direct_list(col,**kwargs):
    rown,val=_parse_args(kwargs)
    with open('databases/batch.csv','r') as csvfile:
        for row in reader(csvfile):
            if row[rown]==val:
                return row[col].split(':')
    return -1
def create_batch(**kwargs):
    with open('databases/batch.csv','a') as csvfile:
        writer(csvfile).writerow([
            kwargs['batch_id'],
            kwargs['batch_name'],
            kwargs['department_name'],
            ':'.join(kwargs['courses']),
            ':'.join(kwargs['students'])
        ])
students=partial(_direct_list,4)
courses=partial(_direct_list,3)
def batch_performance(**kwargs):
    rown,val=_parse_args(kwargs)
    students=[];exams=[]
    with open('databases/batch.csv','r') as csvfile:
        for row in reader(csvfile):
            if row[rown]==val:
                students=row[4].split(':')
                exams=row[3].split(':')
                break
    if not students and not exams:return -1
    lexams=len(exams)
    with open('databases/student.csv','r') as
     ↪  studentcsv,open('databases/course.csv') as csvfile:
        courses=reader(csvfile)
        for row in reader(studentcsv):
            student_id=row[0]
            if student_id in students:
                total=0
                for course in courses:
                    if course[0] in exams:
                        marks=course[2]
                        i=marks.index(student_id)
                        a=marks.find(':',i)
```

```
                    b=marks.find('-',i)
                    total+=float(marks[a+1:b])
            csvfile.seek(0)
            yield Student(row[2],row[1],total/lexams)
def batch_statistics(**kwargs):
    slices,roll_numbers=[],[]
    for student in batch_performance(**kwargs):
        slices.append(student.percentage)
        roll_numbers.append(student.roll)
    name=tuple(kwargs.values())[0]
    title(name)
    xticks([],[])
    yticks([],[])
    style.use('Solarize_Light2')
    pie(slices,labels=roll_numbers,shadow=True,frame=True)
    savefig(f'outputs/Batch Statistics-{name}.png')
    close()
```

department.py

```
from csv import reader,writer
from collections import namedtuple
from matplotlib.pyplot import
 ↪  plot,xlabel,ylabel,style,title,close,savefig
Batch=namedtuple('Performance',('batch','average'))
def _parse_args(argdict):
    wrong_arg=Exception('Either provide department_id or
     ↪  department_name')
    if len(argdict)>1:raise wrong_arg
    (param,val),=argdict.items()
    if param=='department_id':rown=0
    elif param=='department_name':rown=1
    else:raise wrong_arg
    return rown,val
def create_department(**kwargs):
    with open('databases/department.csv','a') as db:
        writer(db).writerow([
            kwargs['department_id'],
            kwargs['department_name'],
            ':'.join(kwargs['batches'])
        ])
def batches(**kwargs):
    rown,val=_parse_args(kwargs)
    with open('databases/department.csv','r') as db:
        for row in reader(db):
            if row[rown]==val:
                return row[2].split(':')
    return -1
def batch_averages(**kwargs):
    with open('databases/batch.csv','r') as
     ↪  batch_csv,open('databases/course.csv','r') as course_csv:
        batch_db=reader(batch_csv)
        course_db=reader(course_csv)
        for batch in batches(**kwargs):
```

```
            total=0
            for row in batch_db:
                if row[0]==batch:
                    batch_csv.seek(0)
                    courses=row[3].split(':')
                    students=row[4].split(':')
                    batch_csv.seek(0)
                    break
            for course in courses:
                for row in course_db:
                    if row[0]==course:
                        performance=row[2]
                        for student in students:
                            i=performance.index(student)
                            a=performance.find(':',i)
                            b=performance.find('-',i)
                            total+=float(performance[a+1:b])
                        course_csv.seek(0)
                        break
            yield Batch(batch,total/(len(students)*len(courses)))
def department_statistics(**kwargs):
    def year(performance):
        a=float(performance.batch[-2:])
        if a>22:
            return 1900+a
        return 2000+a
    stat=list(batch_averages(**kwargs))
    stat.sort(key=year)
    style.use('Solarize_Light2')
    plot([p.average for p in stat],[p.batch for p in
     ↪  stat],linestyle='--')
    xlabel('Batch Average')
    ylabel('Batch')
    name=tuple(kwargs.values())[0]
    title(name)
    savefig(f'outputs/Department Statistics-{name}')
    close()
```

examination.py

```
from csv import reader,writer
from numpy import nan,linspace
from collections import namedtuple
from matplotlib.pyplot import
 ↪  scatter,title,xlabel,ylabel,style,legend,close,savefig
from matplotlib.cm import Oranges as colormap #change to change
 ↪  colormap
Student=namedtuple('Performance',('student_id','average'))
class Examination:
    def __init__(self,*batches):
        self.name=input('Name of examination : ')
        exam_data={}
        course_name={}
        #remember data
        with open('databases/course.csv','r') as csvfile:
            csvfile.readline()
```

```
        for course_id,name,performance in reader(csvfile):
            exam_data[course_id]={} if performance=='' else
                dict((i.split(':') for i in
                ↪  performance.split('-')))
            course_name[course_id]=name
    self.batches=batches
    plot_data={}
    #input data
    self.student_performance=[]
    with open('databases/batch.csv','r') as
    ↪  batchcsv,open('databases/student.csv') as studentcsv:
        student_info=reader(studentcsv)
        for row in reader(batchcsv):
            batch_id=row[0]
            if batch_id in batches:
                print(batch_id)
                courses=row[3].split(':')
                lcourses=len(courses)
                students=row[4].split(':')
                lstudents=len(students)
                for student in students:
                    total=0
                    for info in student_info:
                        if info[0]==student:#found student id
                            print(f'\t{info[2]}')#print roll
                            ↪  number
                            studentcsv.seek(0)
                            break
                    for course in courses:
                        entered=input(f'\t\t{course}: ')
                        marks=0 if entered=='' else float(entered)
                        total+=marks
                        exam_data[course][student]=marks
                        try:
                            plot_data[course][batch_id]+=marks/(l
                            ↪  courses*lstudents)
                        except KeyError:
                            try:
                                plot_data[course][batch_id]=marks
                                ↪  /(lcourses*lstudents)
                            except KeyError:
                                plot_data[course]={batch_id:marks
                                ↪  /(lcourses*lstudents)}
                    self.student_performance.append(Student(stude
                    ↪  nt,total/lcourses))
    #save data
    with open('databases/course.csv','w') as csvfile:
        db=writer(csvfile)
        db.writerow(['Course ID','Course Name','Marks Obtained'])
        for course in course_name:
            db.writerow([
                course,
                course_name[course],
                '-'.join((f'{student}:{marks}' for student,marks
                ↪  in exam_data[course].items()))
            ])
    #arrange data
```

```
        self.data=[]
        self.courses=[]
        for course,course_data in plot_data.items():
            batch_data=[]
            for batch in batches:
                try:
                    batch_data.append(course_data[batch])
                except KeyError:
                    batch_data.append(nan)
            self.courses.append(course)
            self.data.append(batch_data)
    def statistics(self):
        style.use('Solarize_Light2')
        xlabel('Average Marks')
        ylabel('Batch')
        title(self.name)
        legend(
            (scatter(marks,self.batches,color=color,edgecolor='black'
                ) for marks,color in
                zip(self.data,colormap(linspace(0,1,len(self.data))))
                ),
            self.courses
        )
        savefig(f'outputs/{self.name} Exam.png')
        close()
```

5 Outputs

Command Line Interface

```
1. Student
2. Course
3. Batch
4. Department
5. Examination
: 1
    1. Create a new student
    2. Update details of a student
    3. Remove a student
    4. Generate report of a student
    : 1
                Student ID: IT9624
                Student Name: Kartik Joshi
                Class Roll No: B-22
                Batch ID: IT96
1. Student
2. Course
3. Batch
4. Department
5. Examination
: 1
    1. Create a new student
    2. Update details of a student
    3. Remove a student
    4. Generate report of a student
    : 2
                Student ID: ECE1029
                Student Name: Kanav Sagar
                Class Roll No: A-43
```

```
1. Student
2. Course
3. Batch
4. Department
5. Examination
: 1
    1. Create a new student
    2. Update details of a student
    3. Remove a student
    4. Generate report of a student
    : 3
                Student ID: IT9266
1. Student
2. Course
3. Batch
4. Department
5. Examination
: 1
    1. Create a new student
    2. Update details of a student
    3. Remove a student
    4. Generate report of a student
    : 4
                Student ID: CSE0466
1. Student
2. Course
3. Batch
4. Department
5. Examination
: 2
    1. Create a new course
    2. View performance of all students
    3. Create course statistics
    : 1
                Course ID: C011
                Course Name: Robotics
                        Class Roll Number: D-56
                        Student Name: Yashvi Khurana
                        Marks: 89
                        Class Roll Number: F-92
                        Student Name: Saksham Khosla
                        Marks: 94
                        Class Roll Number:
1. Student
2. Course
3. Batch
4. Department
5. Examination
: 2
    1. Create a new course
    2. View performance of all students
    3. Create course statistics
    : 2
                Course: Mechanics
                         Student(roll='C-63', name='Anya Chawla',
                          ↪  marks=84.0)
                         Student(roll='E-00', name='Gatik Hayer',
                          ↪  marks=96.0)
                         Student(roll='G-08', name='Faiyaz Tella',
                          ↪  marks=79.0)
                         Student(roll='B-61', name='Neysa Tandon',
                          ↪  marks=73.0)
                         Student(roll='A-20', name='Yashvi Acharya',
                          ↪  marks=78.0)
                         Student(roll='B-69', name='Zara Badal',
                          ↪  marks=67.0)
```

```
                        Student(roll='H-95', name='Stuvan Devi',
                        ↪  marks=96.0)
                        Student(roll='C-34', name='Samarth Agarwal',
                        ↪  marks=100.0)
                        Student(roll='A-82', name='Hiran Mandal',
                        ↪  marks=73.0)
                        Student(roll='C-71', name='Shlok Bir',
                        ↪  marks=87.0)
                        Student(roll='F-62', name='Prerak Raju',
                        ↪  marks=84.0)
                        Student(roll='F-18', name='Siya Hegde',
                        ↪  marks=80.0)
                        Student(roll='F-84', name='Jivin Varty',
                        ↪  marks=77.0)
                        Student(roll='G-25', name='Mahika Golla',
                        ↪  marks=34.0)
                        Student(roll='C-29', name='Priyansh Walia',
                        ↪  marks=71.0)
                        Student(roll='D-74', name='Hazel Gala',
                        ↪  marks=67.0)
                        Student(roll='C-58', name='Hridaan Bains',
                        ↪  marks=44.0)
1. Student
2. Course
3. Batch
4. Department
5. Examination
: 2
    1. Create a new course
    2. View performance of all students
    3. Create course statistics
    : 3
                Course: C006
1. Student
2. Course
3. Batch
4. Department
5. Examination
: 3
    1. Create a new batch
    2. View list of students in a batch
    3. View list of courses taught in a batch
    4. View performance of a batch
    5. Create pie chart of percentage of all students
    : 1
                Batch ID: IT22
                Batch Name: IT 2022-2026
                Department Name: IT
                        Enter the courses for IT22
                                : C005
                                : C010
                                :
                        Enter the students for IT22
                                : IT2245
                                : IT2234
                                : IT2289
                                :
1. Student
2. Course
3. Batch
4. Department
5. Examination
: 3
    1. Create a new batch
```

```
    2. View list of students in a batch
    3. View list of courses taught in a batch
    4. View performance of a batch
    5. Create pie chart of percentage of all students
    : 2
                Batch ID: IT12
                 ['IT1206', 'IT1231']
1. Student
2. Course
3. Batch
4. Department
5. Examination
: 3
    1. Create a new batch
    2. View list of students in a batch
    3. View list of courses taught in a batch
    4. View performance of a batch
    5. Create pie chart of percentage of all students
    : 3
                Batch ID: CSS_E04_5
                          ['C001', 'C002', 'C003', 'C006', 'C008',
                           ↪  'C009']
1. Student
2. Course
3. Batch
4. Department
5. Examination
: 3
    1. Create a new batch
    2. View list of students in a batch
    3. View list of courses taught in a batch
    4. View performance of a batch
    5. Create pie chart of percentage of all students
    : 4
                Batch ID: CSE92
                          Student(roll='G-66', name='Misha Ghose',
                           ↪  percentage=66.0)
                          Student(roll='C-30', name='Onkar Baral',
                           ↪  percentage=74.16666666666667)
1. Student
2. Course
3. Batch
4. Department
5. Examination
: 3
    1. Create a new batch
    2. View list of students in a batch
    3. View list of courses taught in a batch
    4. View performance of a batch
    5. Create pie chart of percentage of all students
    : 5
                Batch ID: ECE15
1. Student
2. Course
3. Batch
4. Department
5. Examination
: 4
    1. Create a new department
    2. View batches of a department
    3. View average performance of batches of a department
    4. Create statistics of a department
    :1
                Department ID: BA
                Department Name: Business Administration
```

```
                    Enter the batches for BA
                            : BA22
                            : BA89
                            : BA04
                            :
1. Student
2. Course
3. Batch
4. Department
5. Examination
: 4
    1. Create a new department
    2. View batches of a department
    3. View average performance of batches of a department
    4. Create statistics of a department
    :2
              Department ID: CSE
                       ['CSE00', 'CSE01', 'CSE02', 'CSE04',
                           'CSE05', 'CSE08', 'CSE09', 'CSE13',
                           'CSE14', 'CSE21', 'CSE91', 'CSE92',
                           'CSE93', 'CSE94', 'CSE95', 'CSE96',
                           'CSE97', 'CSE98']
1. Student
2. Course
3. Batch
4. Department
5. Examination
: 4
    1. Create a new department
    2. View batches of a department
    3. View average performance of batches of a department
    4. Create statistics of a department
    :3
              Department ID: ECE
Performance(batch='ECE02', average=69.0)
Performance(batch='ECE05', average=63.0)
Performance(batch='ECE06', average=57.0)
Performance(batch='ECE10', average=61.2)
Performance(batch='ECE11', average=66.4)
Performance(batch='ECE13', average=76.8)
Performance(batch='ECE15', average=69.6)
Performance(batch='ECE19', average=66.0)
Performance(batch='ECE90', average=80.0)
Performance(batch='ECE97', average=73.6)
Performance(batch='ECE98', average=74.4)
Performance(batch='ECE99', average=6.6)
1. Student
2. Course
3. Batch
4. Department
5. Examination
: 4
    1. Create a new department
    2. View batches of a department
    3. View average performance of batches of a department
    4. Create statistics of a department
    :4
              Department ID: CSE
1. Student
2. Course
3. Batch
4. Department
5. Examination
: 5
```

```
    Hold an examination:
                        Enter the batches for exam
                                : IT12
                                : ECE99
                                :
Name of examination : Mid Semester
ECE99
        D-39
                C001: 89
                C004: 45
                C006: 68
                C008: 74
                C009: 63
IT12
        B-61
                C005: 87
                C010: 96
        A-20
                C005: 74
                C010: 96
    1. View student perfomance in the examination
    2. Create examination statistics
    : 1
[Performance(student_id='ECE9983', average=67.8),
    Performance(student_id='IT1206', average=91.5),
 ⇉  Performance(student_id='IT1231', average=85.0)]
1. Student
2. Course
3. Batch
4. Department
5. Examination
: 5
    Hold an examination:
                        Enter the batches for exam
                                : IT12
                                : ECE99
                                :
Name of examination : End Semester
ECE99
        D-39
                C001: 78
                C004: 89
                C006: 46
                C008: 23
                C009: 78
IT12
        B-61
                C005: 94
                C010: 86
        A-20
                C005: 74
                C010: 19
    1. View student perfomance in the examination
    2. Create examination statistics
    : 2
1. Student
2. Course
3. Batch
4. Department
5. Examination
:
```

CSE0466-report_card.txt

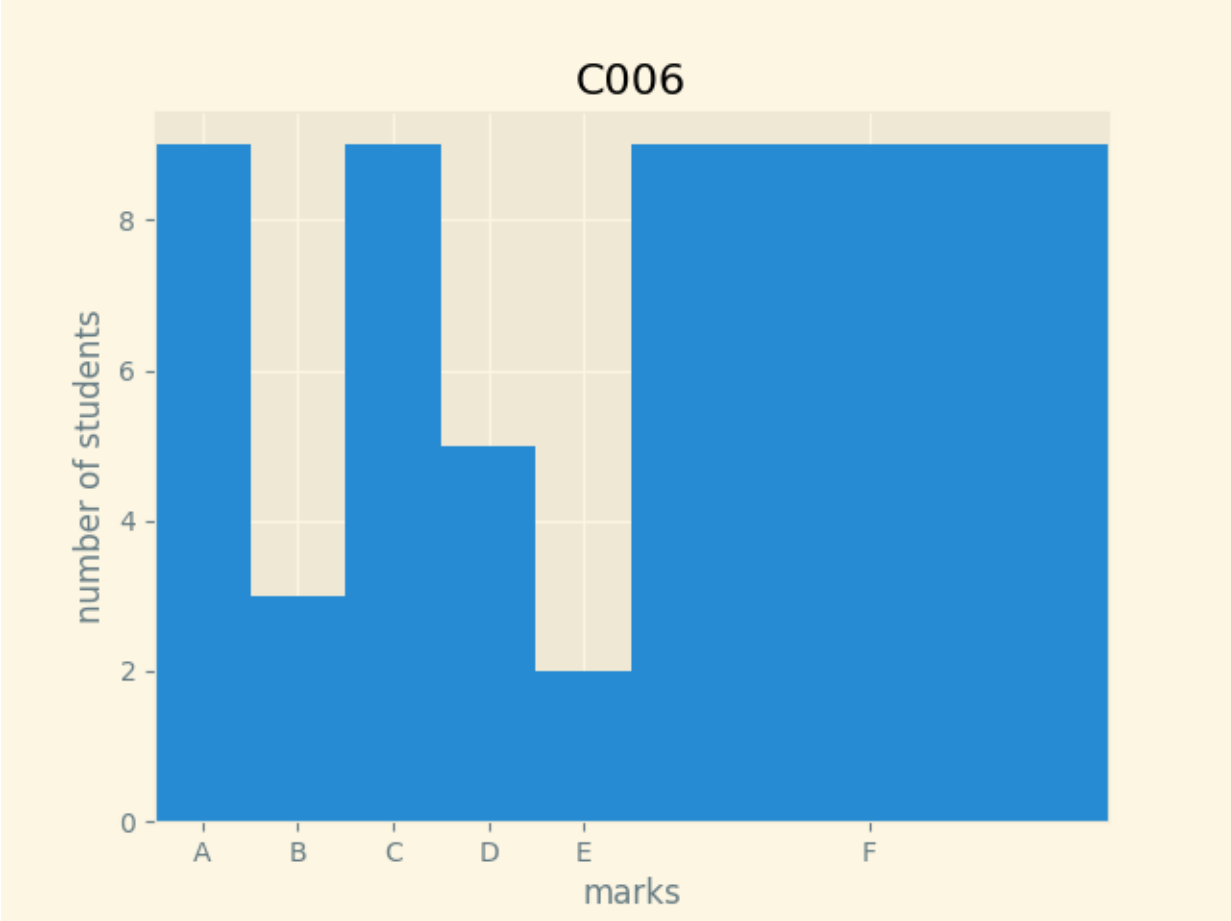
```
Ira Chandran (C-94)

+-----------------+----------+---------------+-----------+------+---------+
```

Course	Course Id	Marks Obtained	Full Marks	Grade	Remarks
Physics	C001	99	100	A	Passed
Mathematics	C002	46	100	F	Failed
Biology	C003	96	100	A	Passed
Python	C006	90	100	A	Passed
Entrepreneurship	C008	68	100	D	Passed
ESP	C009	49	100	F	Failed
Total	-	448	600	C	Passed

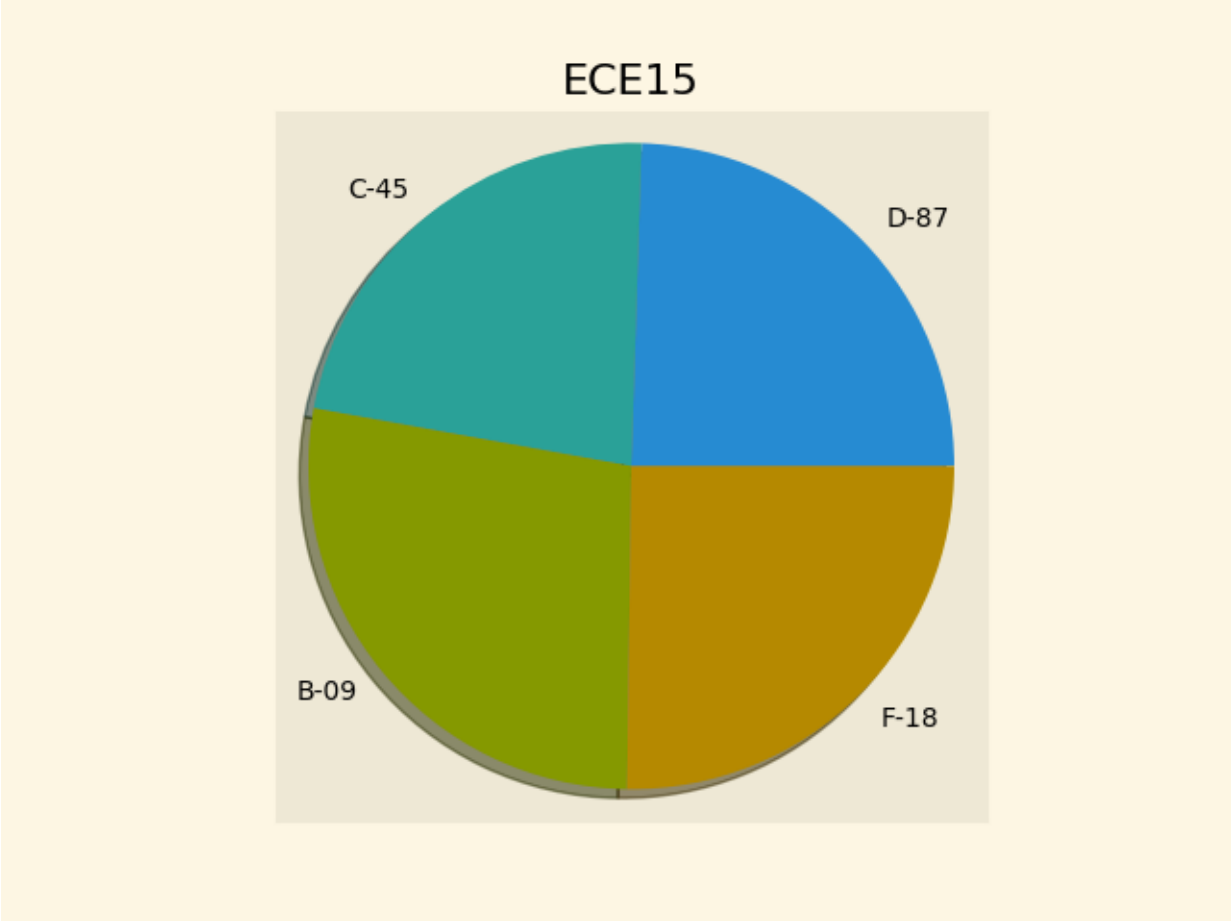
ID:CSE0466

Batch:CSE04

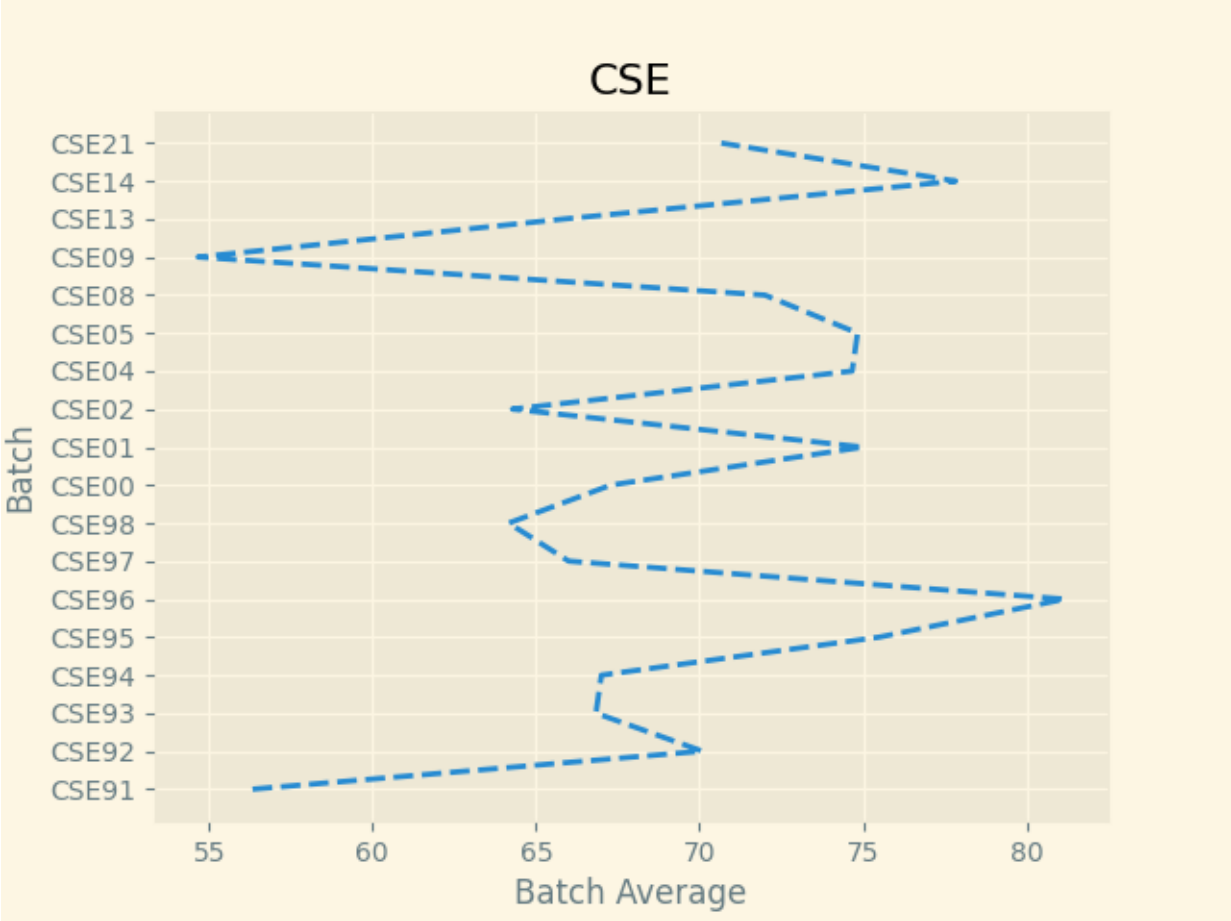
Course Statistics-C006.png



Batch Statistics-ECE15.png



Department Statistics-CSE.png



End Semester Exam.png