OOP Assignment 3:

23K2001

M. Muzammil Siddiqui

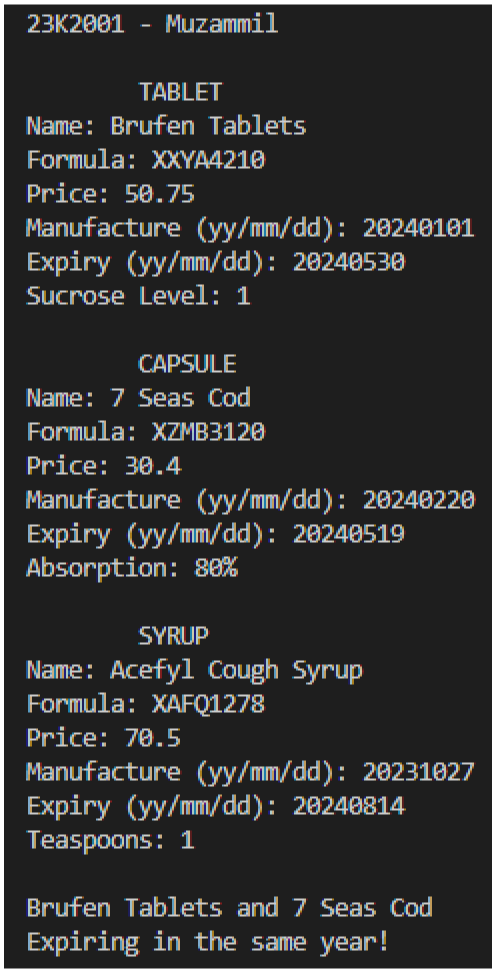
BCS-2J

Q1

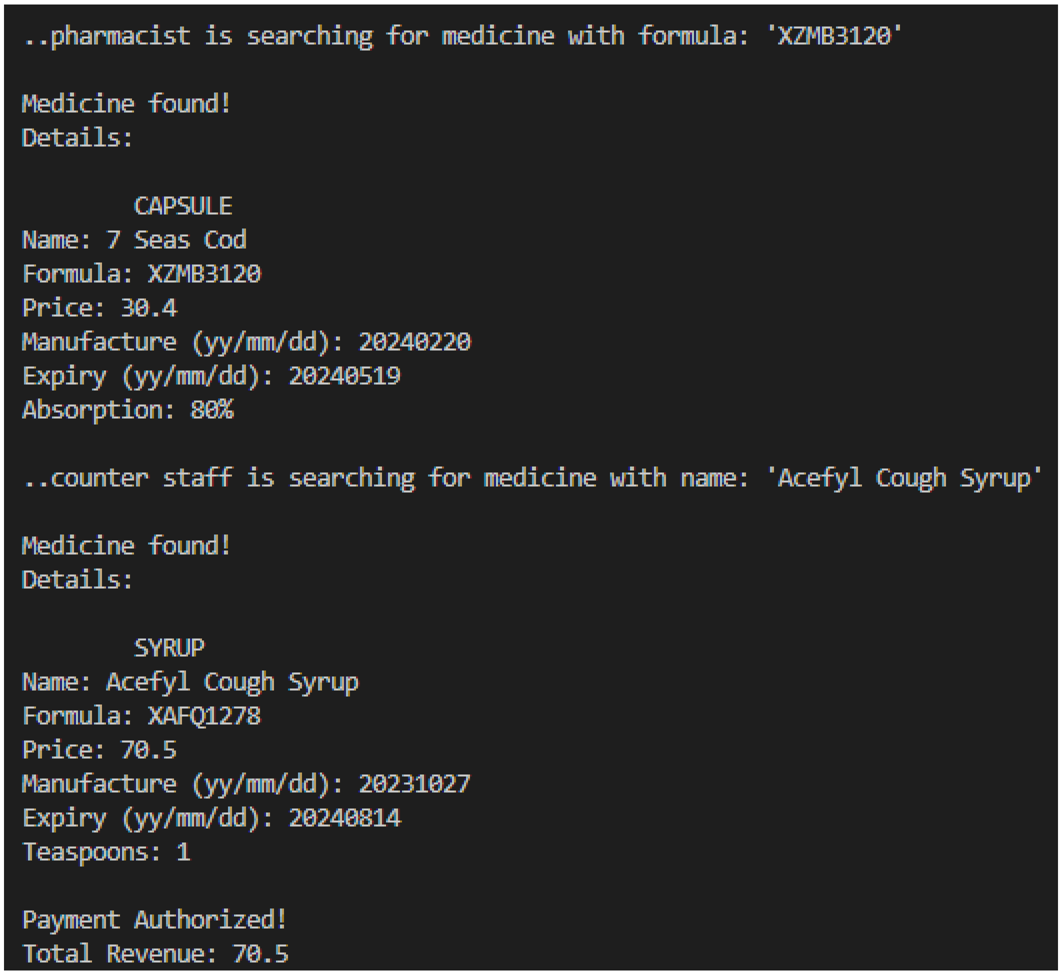
```
//23K2001 Muzammil Q1
#include<iostream>
#include<windows.h>
using namespace std;
static float revenue = 0;
class medicine{
    protected:
    string name,formula;
    float price;
    int mfDate,expDate;

    public:
    medicine(){}
    medicine(string name,string formula,float price,int mfDate,int
expDate){
        this->name=name;
        this->formula=formula;
        this->price=price;
        this->mfDate=mfDate;
        this->expDate=expDate;
    }
    void setName(string name){ this->name = name; }
    void setFormula(string formula){ this->formula = formula; }
    void setPrice(float price){ this->price = price; }
    void setMdate(int mfDate){ this->mfDate = mfDate; }
    void setEDate(int expDate){ this->expDate = expDate; }

    string getName(){ return name; }
    string getFormula(){ return formula; }
    float getPrice(){ return price; }
    int getMdate(){ return mfDate; }
    int getEdate(){ return expDate; }

    virtual void display(){
        cout<<"\nName: "<<name<<endl;
        cout<<"Formula: "<<formula<<endl;
        cout<<"Price: "<<price<<endl;
        cout<<"Manufacture (yy/mm/dd): "<<mfDate<<endl;
```

```cpp
        cout<<"Expiry (yy/mm/dd): "<<expDate<<endl;
    }

    bool operator ==(medicine &m1){
        if(this->expDate/10000==m1.expDate/10000) //20243001 (yymmdd)
            return 1;
        else
            return 0;
    }
};
class Tablet : public medicine{
    protected:
    bool sucroseLevel;

    public:
    Tablet(string n,string f,float p,int mDate,int eDate,bool
s_level):medicine(n,f,p,mDate,eDate),sucroseLevel(s_level){}
    void display() override{
        cout<<"\n\tTABLET";
        medicine::display();
        cout<<"Sucrose Level: "<<sucroseLevel<<endl;
    }
};
class Capsule : public medicine{
    protected:
    int absPerc;

    public:
    Capsule(string n,string f,float p,int mDate,int eDate,int
per):medicine(n,f,p,mDate,eDate),absPerc(per){}
    void display() override{
        cout<<"\n\tCAPSULE";
        medicine::display();
        cout<<"Absorption: "<<absPerc<<"%"<<endl;
    }
};
class Syrup : public medicine{
    protected:
    int teaspoons;
    public:
    Syrup(string n,string f,float p,int mDate,int eDate,int
teaspoon):medicine(n,f,p,mDate,eDate),teaspoons(teaspoon){}
    void display() override{
        cout<<"\n\tSYRUP";
        medicine::display();
        cout<<"Teaspoons: "<<teaspoons<<endl;
    }
};
class pharmacist{
    public:
    void searchMedicine(medicine *meds[],int n,string f){
        cout<<endl<<"..pharmacist is searching for medicine with formula:
'"<<f<<"'"<<endl;
        Sleep(3500);
        int x=1;
        for(int i=0;i<n;i++){
            if(meds[i]->getFormula()==f){
            cout<<endl<<"Medicine found!"<<endl<<"Details: "<<endl;
            meds[i]->display();
            x=0;
            break;
            }
        }
        if(x)
        cout<<"No medicine found with the formula provided!"<<endl;
    }
};
class counter{
    public:
    void searchMedicine(medicine *meds[],int n,string name){
        cout<<endl<<"..counter staff is searching for medicine with name:
'"<<name<<"'"<<endl;
        Sleep(3500);
        int x=1;
        for(int i=0;i<n;i++){
            if(meds[i]->getName()==name){
            cout<<endl<<"Medicine found!"<<endl<<"Details: "<<endl;
            meds[i]->display();
            x=0;
            break;
            }
        }
        if(x)
        cout<<"No medicine found with the name provided!"<<endl;
    }
    void updateRevenue(medicine m){
        revenue+=m.getPrice();
        cout<<endl<<"Payment Authorized!"<<endl;
        cout<<"Total Revenue: "<<revenue<<endl;
    }
};
int main(){
    cout<<"23K2001 - Muzammil"<<endl;
    Tablet brufen("Brufen Tablets","XXYA4210",50.75,20240101,20240530,1);
    Capsule sevenseas("7 Seas Cod","XZMB3120",30.40,20240220,20240519,80);
    Syrup acefyl("Acefyl Cough
Syrup","XAFQ1278",70.5,20231027,20240814,1);

    brufen.display();
    sevenseas.display();
    acefyl.display();

    cout<<endl<<brufen.getName()<<" and "<<sevenseas.getName();
    if(brufen==sevenseas)
        cout<<"\nExpiring in the same year!"<<endl;
    else
        cout<<"\nExpiry year different!"<<endl;

    pharmacist p;
    counter c;
    medicine *supplies[] = {&brufen,&sevenseas,&acefyl};
    p.searchMedicine(supplies,3,"XZMB3120");
    c.searchMedicine(supplies,3,"Acefyl Cough Syrup");
    c.updateRevenue(acefyl);
    return 0;
}
```

Outputs for Q1



```
23K2001 - Muzammil

        TABLET
Name: Brufen Tablets
Formula: XXYA4210
Price: 50.75
Manufacture (yy/mm/dd): 20240101
Expiry (yy/mm/dd): 20240530
Sucrose Level: 1

        CAPSULE
Name: 7 Seas Cod
Formula: XZMB3120
Price: 30.4
Manufacture (yy/mm/dd): 20240220
Expiry (yy/mm/dd): 20240519
Absorption: 80%

        SYRUP
Name: Acefyl Cough Syrup
Formula: XAFQ1278
Price: 70.5
Manufacture (yy/mm/dd): 20231027
Expiry (yy/mm/dd): 20240814
Teaspoons: 1

Brufen Tablets and 7 Seas Cod
Expiring in the same year!
```

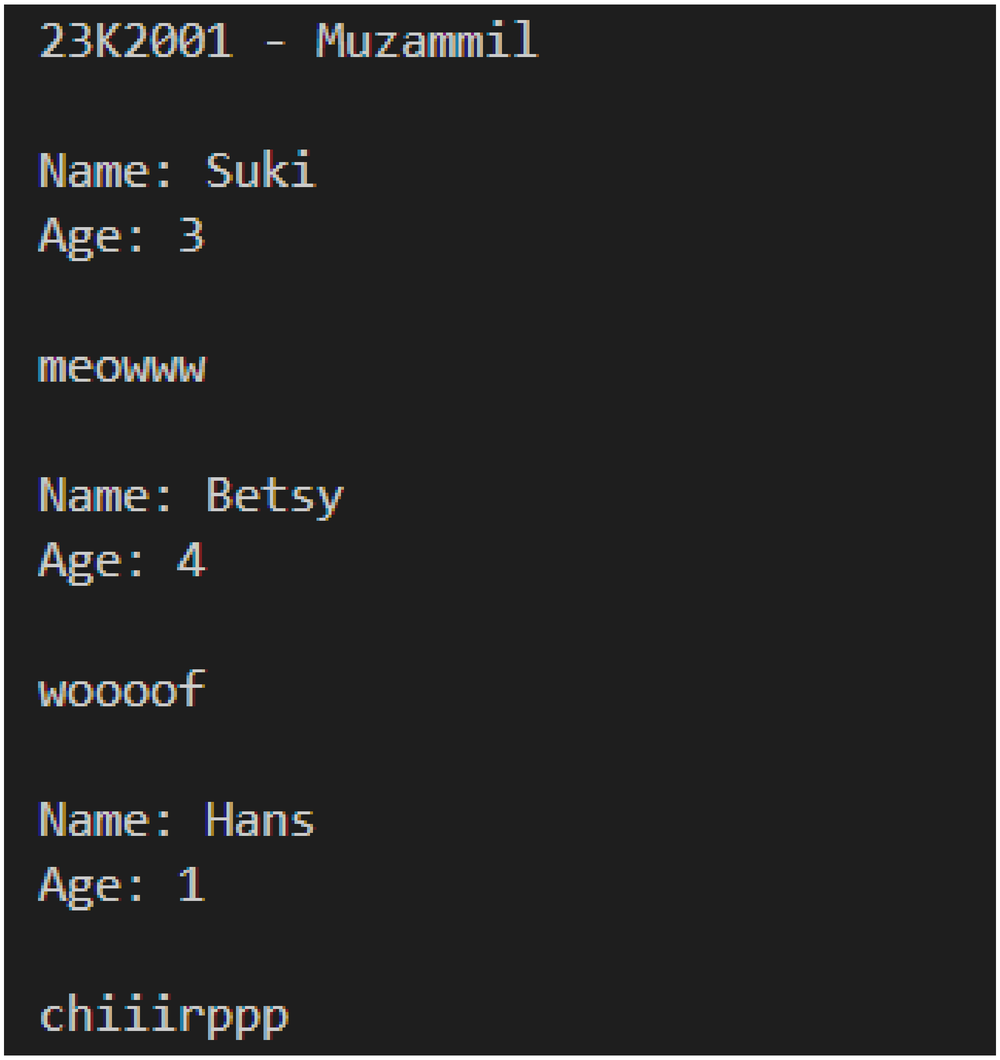


```
..pharmacist is searching for medicine with formula: 'XZMB3120'

Medicine found!
Details:

        CAPSULE
Name: 7 Seas Cod
Formula: XZMB3120
Price: 30.4
Manufacture (yy/mm/dd): 20240220
Expiry (yy/mm/dd): 20240519
Absorption: 80%

..counter staff is searching for medicine with name: 'Acefyl Cough Syrup'

Medicine found!
Details:

        SYRUP
Name: Acefyl Cough Syrup
Formula: XAFQ1278
Price: 70.5
Manufacture (yy/mm/dd): 20231027
Expiry (yy/mm/dd): 20240814
Teaspoons: 1

Payment Authorized!
Total Revenue: 70.5
```

Q2:

```cpp
//23K2001 Muzammil Q2
#include<iostream>
using namespace std;
template <class TN> class Pet{
    protected:
    TN name;
    int age;

    public:
    Pet(TN name,int age):name(name),age(age){}
    virtual void makeSound()=0;
    void display(){
        cout<<"\nName: "<<name<<endl;
        cout<<"Age: "<<age<<endl;
    }
};
class Cat : public Pet<string>{
    public:
    Cat(string n,int a):Pet(n,a){}
    void makeSound(){ cout<<"\nmeowww"<<endl; }
};
class Dog : public Pet<string>{
    public:
    Dog(string n,int a):Pet(n,a){}
    void makeSound(){ cout<<"\nwoooof"<<endl; }
};
class Bird : public Pet<string>{
    public:
    Bird(string n,int a):Pet(n,a){}
    void makeSound(){ cout<<"\nchiiirppp"<<endl; }
};
int main(){
    cout<<"23K2001 - Muzammil"<<endl;

    Cat c1("Suki", 3);
    Dog d1("Betsy", 4);
    Bird b1("Hans", 1);
    Pet<string> *pets[3] = {&c1, &d1,&b1};
    for (int i = 0; i<3; i++){
        pets[i]->display();
        pets[i]->makeSound();
    }
    return 0;
}
```

Outputs for Q2



```
23K2001 - Muzammil

Name: Suki
Age: 3

meowww

Name: Betsy
Age: 4

wooooof

Name: Hans
Age: 1

chiiirppp
```

Q3:

```
//23K2001 Muzammil Q3
#include<iostream>
using namespace std;
template<class T> class matrix{
    protected:
    int rows,cols;
    T **mat=nullptr;

    public:
    matrix(int r,int c):rows(r),cols(c){
        mat = new T*[rows];
        for(int i=0;i<rows;i++){
            mat[i] = new T[cols];
            for(int j=0;j<cols;j++)
            mat[i][j] = 0;
        }
    }
    matrix(const matrix &m){
        rows = m.getRows();
        cols = m.getCols();
        mat = new T*[rows];
        for(int i=0;i<rows;i++){
            mat[i] = new T[cols];
            for(int j=0;j<cols;j++)
            mat[i][j] = m.getV(i,j);
        }
    }

    void setV(int r,int c, T value){ mat[r][c] = value; }
    T getV(int r,int c) const { return mat[r][c]; }
    int getRows() const { return rows; }
    int getCols() const { return cols; }

    matrix operator +(matrix &m1) const {
        if(this->rows==m1.getRows() && this->cols==m1.getCols()){
            matrix sum(this->getRows(),this->getCols());
            for(int i=0;i<this->getRows();i++){
                for(int j=0;j<this->getCols();j++){
                    sum.setV(i,j,this->getV(i,j)+m1.getV(i,j));
                }
            }
            return sum;
        }
        else{
        cout<<"\nSorry cannot add! (Different orders)"<<endl;
        return matrix(0,0);
        }
    }
    matrix operator -(matrix &m1) const {
        if(this->rows==m1.getRows() && this->cols==m1.getCols()){
            matrix sub(this->getRows(),this->getCols());
            for(int i=0;i<this->rows;i++){
                for(int j=0;j<this->cols;j++)
                    sub.setV(i,j,this->getV(i,j)-m1.getV(i,j));
            }
            return sub;
        }
        else{
        cout<<"\nSorry cannot subtract! (Different orders)"<<endl;
        return matrix(0,0);
        }
    }
    matrix operator *(matrix& m1) const {
        if(this->cols==m1.getRows()){
            matrix prod(this->rows,m1.getCols());
            for(int i=0;i<this->rows;i++){
                for(int j=0;j<prod.getCols();j++){
                    for(int x=0;x<this->cols;x++)

prod.setV(i,j,prod.getV(i,j)+(this->getV(i,x)*m1.getV(x,j)));
                }
            }
            return prod;
        }
        else{
        cout<<"\nSorry cannot multiply! (Orders not compatible)"<<endl;
        return matrix(0,0);
        }
    }

    virtual void show(){ // cannot make pure virtual as operator
overloadings won't work
            if(getRows()==0 && getCols()==0)
        cout<<"error displaying matrix"<<endl;
        else{
                for(int i=0;i<rows;i++){
                    for(int j=0;j<cols;j++)
                        cout<<mat[i][j]<<" ";
                    cout<<endl;
                }
            }
        }

    ~matrix(){
        for(int i=0;i<rows;i++)
        delete[] mat[i];

        delete[] mat;
    }
};

class matrixInt : public matrix<int>{
public:
matrixInt(int r,int c):matrix(r,c){}
matrixInt(const matrix &m): matrix(m){
        rows = m.getRows();
        cols = m.getCols();
        mat = new int*[rows];
        for(int i=0;i<rows;i++){
            mat[i] = new int[cols];
            for(int j=0;j<cols;j++)
            mat[i][j] = m.getV(i,j);
        }
    }
void show(){
    if(getRows()==0 && getCols()==0)
    cout<<"error displaying matrix"<<endl;
    else{
```

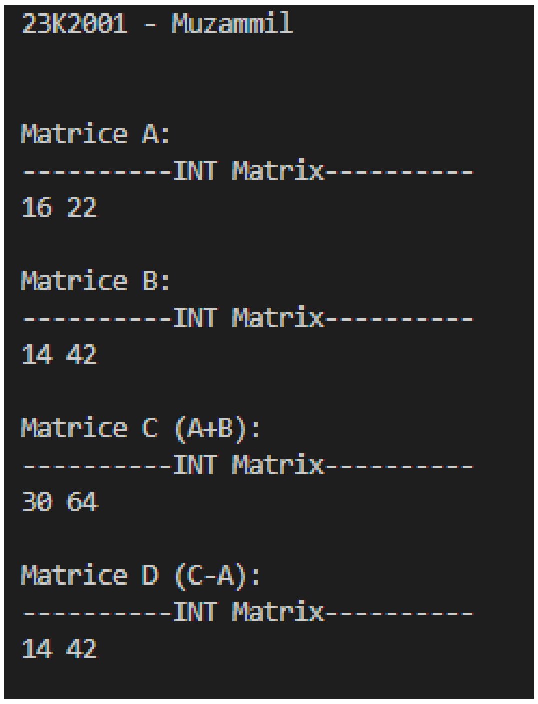
```cpp
        cout<<"----------INT Matrix----------"<<endl;
            for(int i=0;i<rows;i++){
                for(int j=0;j<cols;j++)
                    cout<<mat[i][j]<<" ";
                cout<<endl;
            }
        }
    }
};

class matrixDouble : public matrix<double>{
public:
matrixDouble(int r,int c):matrix(r,c){}
matrixDouble(const matrix &m): matrix(m){
        rows = m.getRows();
        cols = m.getCols();
        mat = new double*[rows];
        for(int i=0;i<rows;i++){
            mat[i] = new double[cols];
            for(int j=0;j<cols;j++)
            mat[i][j] = m.getV(i,j);
        }
    }
void show(){
    if(getRows()==0 && getCols()==0)
    cout<<"error displaying matrix"<<endl;
    else{
        cout<<"----------DOUBLE Matrix----------"<<endl;
            for(int i=0;i<rows;i++){
                for(int j=0;j<cols;j++)
                    cout<<mat[i][j]<<" ";
                cout<<endl;
            }
        }
    }
};
int main(){
    cout<<"23K2001 - Muzammil\n\n"<<endl;
    matrixInt A(1,2);
    matrixInt B(1,2);
```

```
A.setV(0,0,16);
A.setV(0,1,22);
B.setV(0,0,14);
B.setV(0,1,42);

cout<<"Matrice A: "<<endl;
A.show();
cout<<endl<<"Matrice B: "<<endl;
B.show();

matrixInt C = A+B;
cout<<endl<<"Matrice C (A+B): "<<endl;
C.show();

matrixInt D = C-A;
cout<<endl<<"Matrice D (C-A): "<<endl;
D.show();

matrixInt E(2,1);
matrixInt F(1,2);
E.setV(0,0,10);
E.setV(1,0,15);
F.setV(0,0,2);
F.setV(0,1,6);

matrixInt G = E*F;
cout<<endl<<"Matrice E: "<<endl;
E.show();
cout<<endl<<"Matrice F: "<<endl;
F.show();
cout<<endl<<"Matrice G (E*F): "<<endl;
G.show();

matrixDouble M(2,2);
matrixDouble N(2,2);
M.setV(0,0,1.5);
M.setV(0,1,3.9);
M.setV(1,0,4.25);
M.setV(1,1,10.8);
```

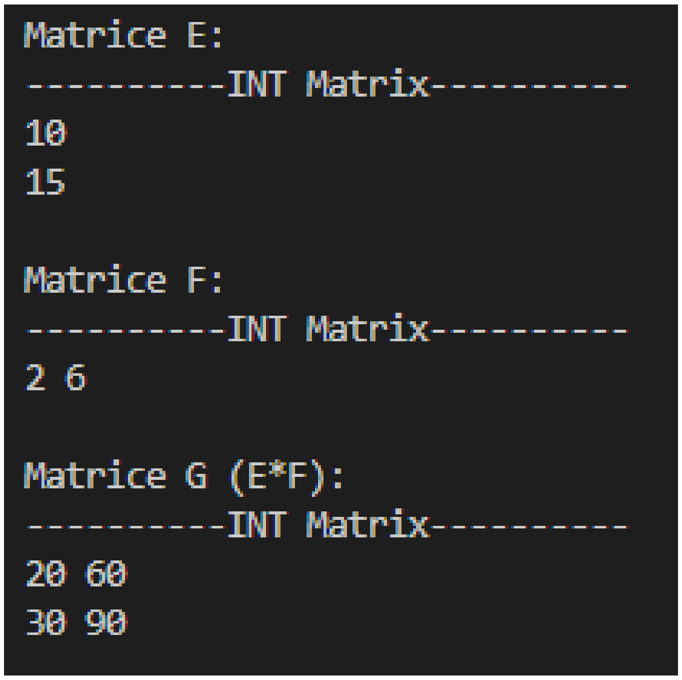
```
    N.setV(0,0,2.5);
    N.setV(0,1,0.3);
    N.setV(1,0,4.5);
    N.setV(1,1,9.11);

    cout<<endl<<"Matrice M: "<<endl;
    M.show();
    cout<<endl<<"Matrice N: "<<endl;
    N.show();

    matrixDouble X = M+N;
    cout<<endl<<"Matrice X (M+N): "<<endl;
    X.show();

    matrixDouble Y = X-M;
    cout<<endl<<"Matrice Y (X-M): "<<endl;
    Y.show();

    matrixDouble Z = M*N;
    cout<<endl<<"Matrice X (M*N): "<<endl;
    Z.show();

    matrixInt o1(1,2);
    matrixInt o2(1,2);
    o1.setV(0,0,1);
    o1.setV(0,1,2);
    o2.setV(0,0,3);
    o2.setV(0,1,4);

    cout<<endl<<"Matrice o1: "<<endl;
    o1.show();
    cout<<endl<<"Matrice o2: "<<endl;
    o2.show();

    cout<<endl<<"Matrix ANS (o1*o2):"<<endl;
    matrixInt ans = o1*o2;
    ans.show();
    return 0;
}
```

Please consider that by making the display function as abstract (of matrix generic class), matrix class becomes abstract which then doesn't allow to instantiate objects or return an object when operator overloading is invoked, so I haven't made it abstract.

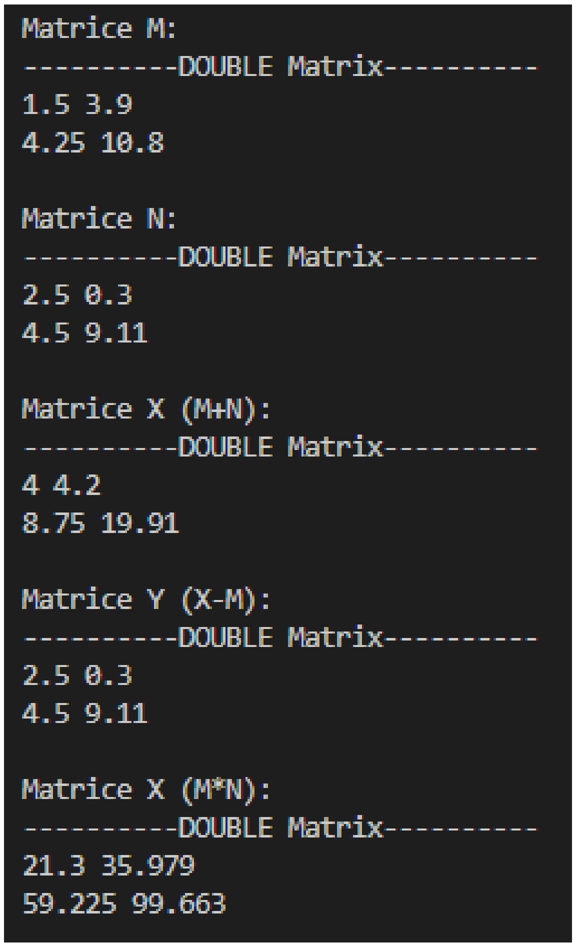
Outputs for Q3



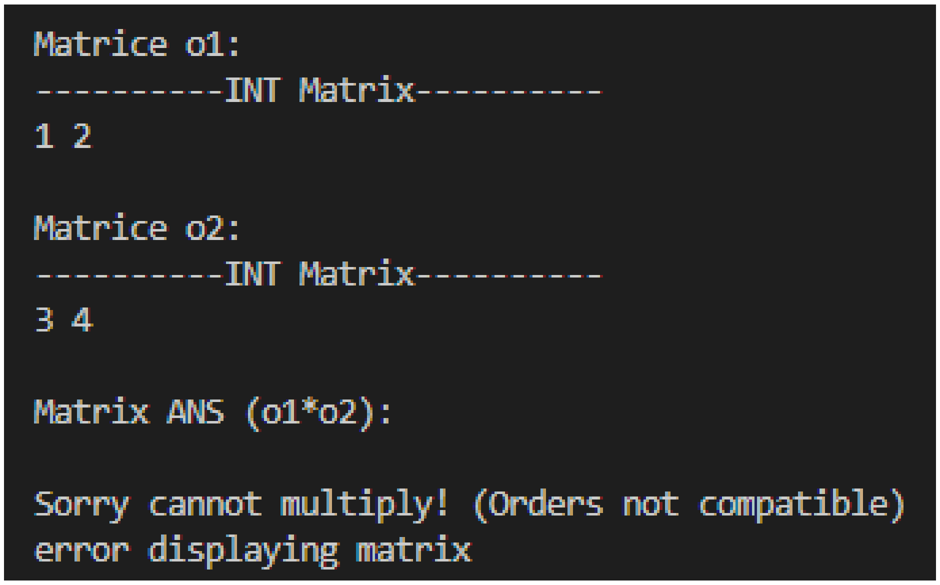
```
23K2001 - Muzammil


Matrice A:
----------INT Matrix----------
16 22

Matrice B:
----------INT Matrix----------
14 42

Matrice C (A+B):
----------INT Matrix----------
30 64

Matrice D (C-A):
----------INT Matrix----------
14 42
```



```
Matrice E:
----------INT Matrix----------
10
15

Matrice F:
----------INT Matrix----------
2 6

Matrice G (E*F):
----------INT Matrix----------
20 60
30 90
```



```
Matrice M:
----------DOUBLE Matrix----------
1.5 3.9
4.25 10.8

Matrice N:
----------DOUBLE Matrix----------
2.5 0.3
4.5 9.11

Matrice X (M+N):
----------DOUBLE Matrix----------
4 4.2
8.75 19.91

Matrice Y (X-M):
----------DOUBLE Matrix----------
2.5 0.3
4.5 9.11

Matrice X (M*N):
----------DOUBLE Matrix----------
21.3 35.979
59.225 99.663
```



```
Matrice o1:
----------INT Matrix----------
1 2

Matrice o2:
----------INT Matrix----------
3 4

Matrix ANS (o1*o2):

Sorry cannot multiply! (Orders not compatible)
error displaying matrix
```

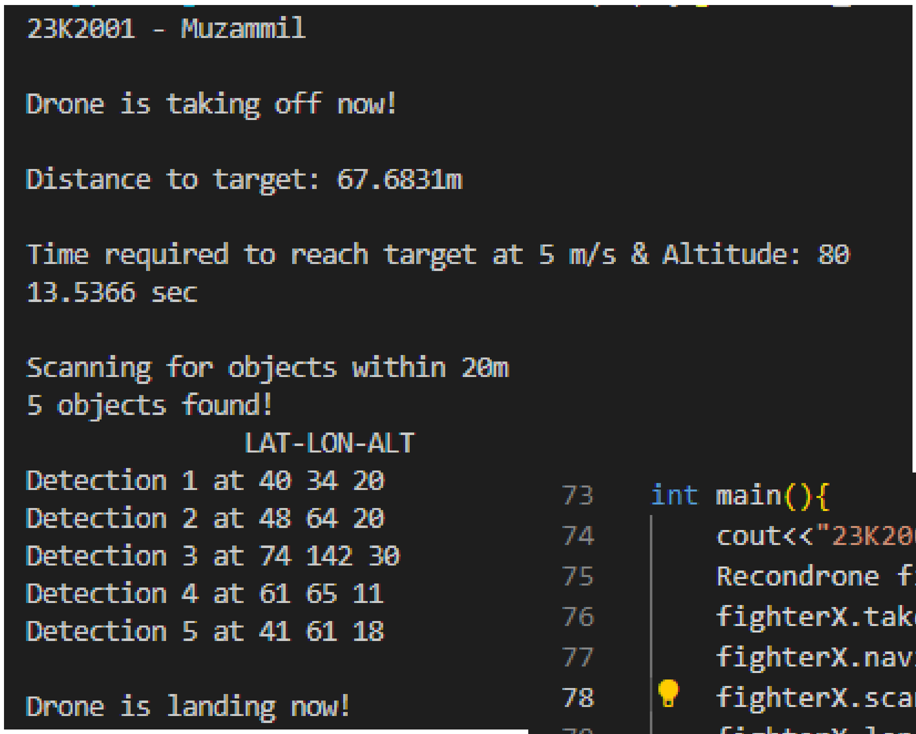
Q4:

```
//23K2001 Muzammil Q4
#include<iostream>
#include <unistd.h>
#include<cmath>
using namespace std;
class Flyable{
    public:
    virtual void takeoff()=0;
    virtual void land()=0;
    virtual void navigate(float latitude,float longitude,float
altitude)=0;
};
class Scannable{
    public:
    virtual void scanArea(float radius)=0;
};
class Drone : public Flyable, public Scannable{
    protected:
    float longit,latit,speed,alt;

    public:
    Drone(float la,float lo,float a,float s):
longit(lo),latit(la),alt(a),speed(s){}
    void adjustAlt(float m){ alt = m; }
    float getAlt(){ return alt; }
    void setSpeed(float s=0){ speed = s; }
    void setPos(float latitude,float longitude){
        latit = latitude;
        longit = longitude;
    }
};
class Recondrone: public Drone{
    protected:
    int cameraResolution, maxTime;

    public:
    Recondrone(float la,float lo,float a,float s,int c,int
m):Drone(la,lo,a,s),cameraResolution(c),maxTime(m){}
```

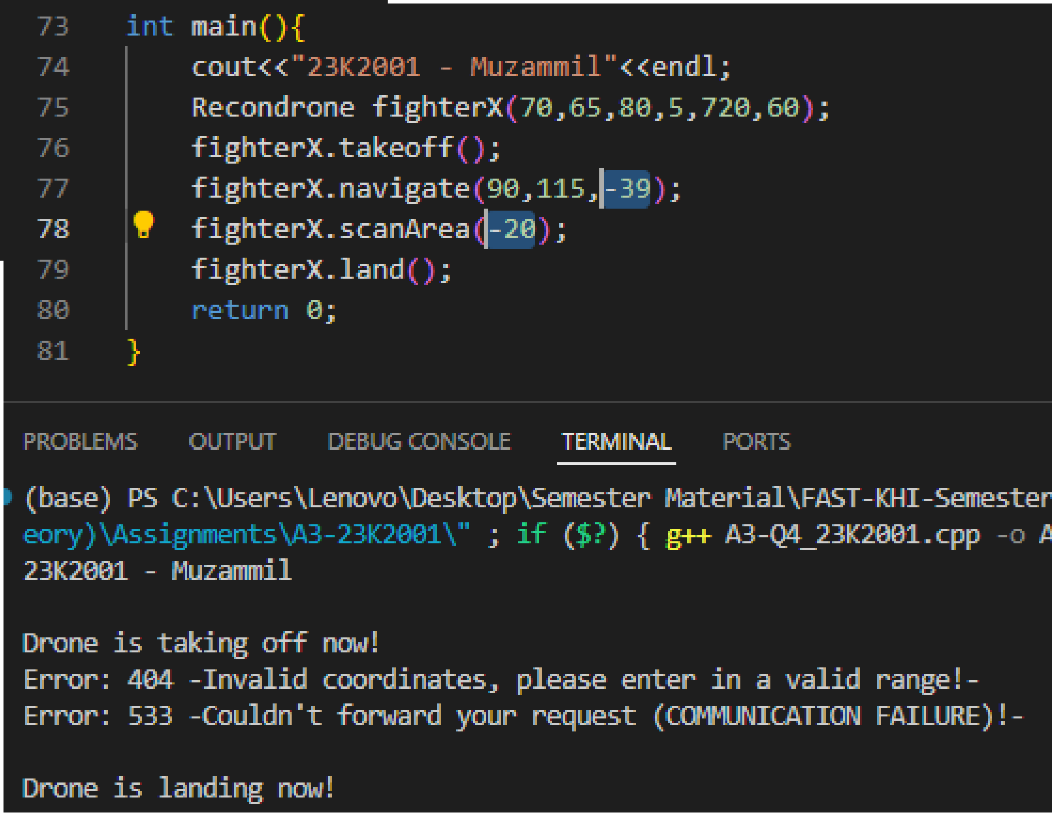
```
    void takeoff(){ cout<<"\nDrone is taking off now!"<<endl; }
    void land(){ cout<<"\nDrone is landing now!"<<endl; }
    void scanArea(float radius){
        try{
            if(radius<0)
            throw 533;

            cout<<"\nScanning for objects within "<<radius<<"m"<<endl;
            int objs = rand()%6;
            cout<<objs<<" objects found!"<<endl;
            cout<<"\t      LAT-LON-ALT"<<endl;
            for(int i=1;i<=objs;i++){
                cout<<"Detection "<<i<<" at ";
                cout<<(float)(rand()%90)<<" "<<(float)(rand()%180)<<"
"<<(float)(rand()%(int)(getAlt()))<<endl;
            }
        }
        catch(int e){ cout<<"Error: "<<e<<" -Couldn't forward your request
(COMMUNICATION FAILURE)!-"<<endl; }
    }

    void navigate(float latitude,float longitude,float altitude){
        try{
            if(latitude>90 || longitude>180 || altitude<0)
            throw 404;

            cout<<"\nDistance to target: ";
            float d =
sqrt(pow((latitude-latit),2)+pow((longitude-longit),2)+pow((altitude-alt),
2));
            cout<<d<<"m"<<endl;
            cout<<"\nTime required to reach target at "<<speed<<" m/s ";
            cout<<"& Altitude: "<<alt<<endl;
            cout<<d/speed<<" sec"<<endl;
            setPos(latitude,longitude);
            adjustAlt(altitude);
        }
        catch(int e){ cout<<"Error: "<<e<<" -Invalid coordinates, please
enter in a valid range!-"<<endl; }
    }
```

```
};
int main(){
    cout<<"23K2001 - Muzammil"<<endl;
    Recondrone fighterX(70,65,80,5,720,60);
    fighterX.takeoff();
    fighterX.navigate(90,115,39);
    fighterX.scanArea(20);
    fighterX.land();
    return 0;
}
```

Outputs for Q4



```
23K2001 - Muzammil

Drone is taking off now!

Distance to target: 67.6831m

Time required to reach target at 5 m/s & Altitude: 80
13.5366 sec

Scanning for objects within 20m
5 objects found!
              LAT-LON-ALT
Detection 1 at 40 34 20
Detection 2 at 48 64 20
Detection 3 at 74 142 30
Detection 4 at 61 65 11
Detection 5 at 41 61 18

Drone is landing now!
```



```
int main(){
    cout<<"23K2001 - Muzammil"<<endl;
    Recondrone fighterX(70,65,80,5,720,60);
    fighterX.takeoff();
    fighterX.navigate(90,115,-39);
    fighterX.scanArea(-20);
    fighterX.land();
    return 0;
}
```

```
(base) PS C:\Users\Lenovo\Desktop\Semester Material\FAST-KHI-Semester
eory)\Assignments\A3-23K2001\" ; if ($?) { g++ A3-Q4_23K2001.cpp -o A
23K2001 - Muzammil

Drone is taking off now!
Error: 404 -Invalid coordinates, please enter in a valid range!-
Error: 533 -Couldn't forward your request (COMMUNICATION FAILURE)!-

Drone is landing now!
```