PHANINDRA S KARANAM
4/27/17

Lab 10 Report

AIM

To become familiar Abstract base class and exception handling

IMPLEMENTATION

Part A

The class baseconic is an abstract base class as it contains pure virtual functions and it is used to inherit the classes Circle and ellipse .

When we try to create a object of ABC the compiler gives an error.

We can create pointers of ABC which point to the derived class in order to dynamically call the function of the base class

All the cases have been verified for code

Part B

Exception handling is used to handle any unwanted results from a given part of the code

The try block is used to designate the part of the code where the exception is expected and the throw statement throws various variables, constant or objects depending on the user specification

The catch statement is used to catch the exception thrown and performs the actions for the necessary exception caught

There may be more than one catch block for one try block

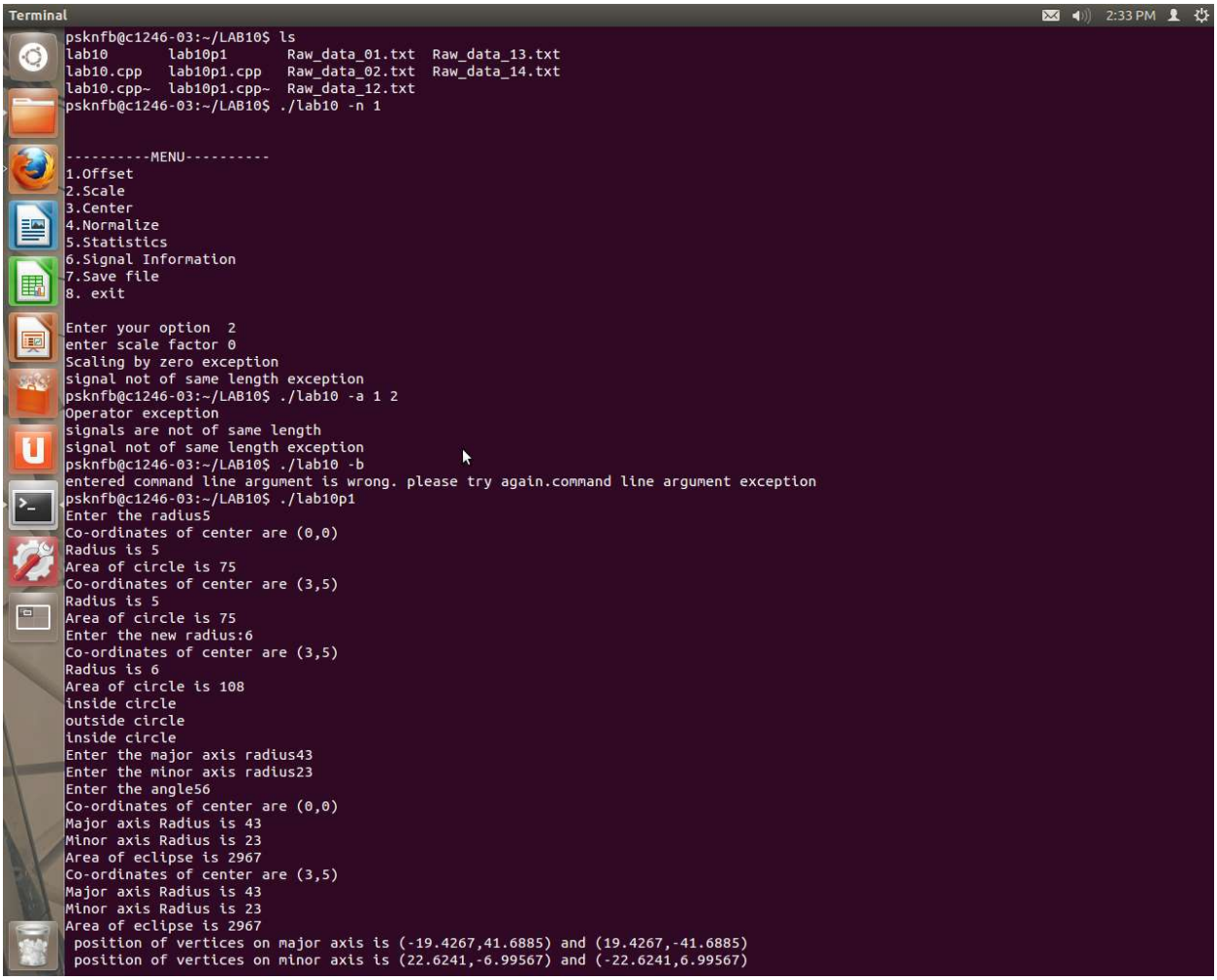
If the throw specification of a function is mismatched by throwing anything other than the specified one the program terminates

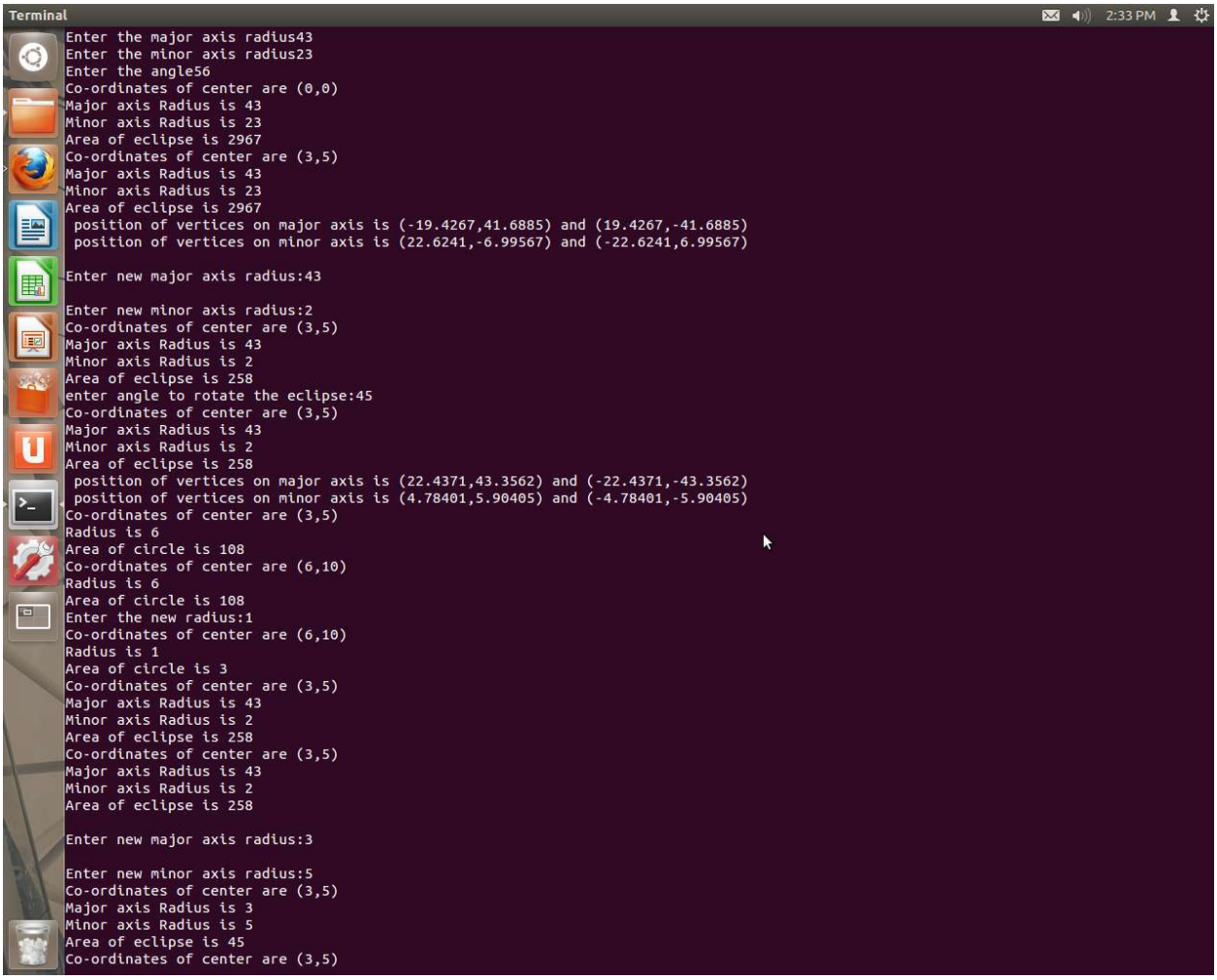
The rethrow is used to check the exception on various levels of the code

The catch(…) i.e. catch all statement catches all the exception irrespective of their type

All the cases have been verified and demonstrated

TESTCASES


Terminal
2:33 PM
psknfb@c1246-03:~/LAB10$ ls
lab10 lab10p1 Raw_data_01.txt Raw_data_13.txt
lab10.cpp lab10p1.cpp Raw_data_02.txt Raw_data_14.txt
lab10.cpp~ lab10p1.cpp~ Raw_data_12.txt
psknfb@c1246-03:~/LAB10$ ./lab10 -n 1
----------MENU----------
1.Offset
2.Scale
3.Center
4.Normalize
5.Statistics
6.Signal Information
7.Save file
8. exit
Enter your option 2
enter scale factor 0
Scaling by zero exception
signal not of same length exception
psknfb@c1246-03:~/LAB10$ ./lab10 -a 1 2
Operator exception
signals are not of same length
signal not of same length exception
psknfb@c1246-03:~/LAB10$ ./lab10 -b
entered command line argument is wrong. please try again.command line argument exception
psknfb@c1246-03:~/LAB10$ ./lab10p1
Enter the radius5
Co-ordinates of center are (0,0)
Radius is 5
Area of circle is 75
Co-ordinates of center are (3,5)
Radius is 5
Area of circle is 75
Enter the new radius:6
Co-ordinates of center are (3,5)
Radius is 6
Area of circle is 108
inside circle
outside circle
inside circle
Enter the major axis radius43
Enter the minor axis radius23
Enter the angle56
Co-ordinates of center are (0,0)
Major axis Radius is 43
Minor axis Radius is 23
Area of eclipse is 2967
Co-ordinates of center are (3,5)
Major axis Radius is 43
Minor axis Radius is 23
Area of eclipse is 2967
position of vertices on major axis is (-19.4267,41.6885) and (19.4267,-41.6885)
position of vertices on minor axis is (22.6241,-6.99567) and (-22.6241,6.99567)



GITHUB LINK

HTTPS://GITHUB.COM/KARANAM97/LAB10

CODE

PART A

```
#include<iostream>
#include<math.h>
using namespace std;

class BaseConic{  // Abstract base class
protected:
double x;
double y;
public:
//BaseConic(double x0 = 0, double y0 = 0){ };
//virtual ~BaseConic(){};
virtual void move(double n_x, double n_y)  = 0;
virtual void resize() = 0;
virtual double Area()  = 0;
virtual void print()  = 0;
};

class circle : public BaseConic {  //Circle class derived from ABC
        private:
        double r;
        public:
        circle();
        circle(double xc, double yc, double rc);
        //~circle();
 void move(double n_x, double n_y) { x = n_x; y = n_y; }
 void resize();
double Area() { return ((double(22/7))*r*r);}
        void position( double x1, double y1);
```

```
 void print();
        };

circle::circle()
{
        x=0;
        y=0;
        cout<<"Enter the radius";
        cin>>r;
}
circle::circle(double xc, double yc, double rc)
{
        x=xc;
        y=yc;
        r=rc;
}
void circle::position(double x1, double y1)
{
        double d;
        d = sqrt(pow((x1-x),2) + pow((y1-y),2));
        if(d<r)
                cout<<"inside circle"<<endl;
        else if (d>r)
                cout<<"outside circle"<<endl;
        else
                cout<<"on the circle"<<endl;
}

void circle::resize() {
        cout<<"Enter the new radius:";
        cin>>r;
```

```
}
void circle::print(){
        cout<<"Co-ordinates of center are ("<<x<<","<<y<<")"<<endl;
        cout<<"Radius is "<<r<<endl;
        cout<<"Area of circle is "<<Area()<<endl;
}

class ellipse : public BaseConic { //Ellipse deived from ABC
        private:
        double a,b,ang;
        public:
        eclipse();
        eclipse(double xc, double yc, double ac, double bc, double angc) { x = xc; y = yc; a = ac; b = bc;
ang = angc;}
        //~eclipse();
   void move(double n_x, double n_y) { x = n_x; y = n_y; }
         void resize();
         double Area() { return ((double (22/7))*a*b);}
        void position();
        void rotate();
 void print();
        };
        eclipse::eclipse()
{
         x=0;
         y=0;
         cout<<"Enter the major axis radius";
         cin>>a;
         cout<<"Enter the minor axis radius";
         cin>>b;
         cout<<"Enter the angle";
```

```
        cin>>ang;
}


        void eclipse::resize()
        {
                cout<<"\nEnter new major axis radius:";
                cin>>a;
                cout<<"\nEnter new minor axis radius:";
                cin>>b;
        }
        void eclipse::position()
        {
                cout<<" position of vertices on major axis is
("<<((a*sin(ang))+x)<<","<<((a*cos(ang))+y)<<") and ("<<-((a*sin(ang))+x)<<","<<-
((a*cos(ang))+y)<<")"<<endl;
                cout<<" position of vertices on minor axis is
("<<((b*cos(ang))+x)<<","<<((b*sin(ang))+y)<<") and ("<<-((b*cos(ang))+x)<<","<<-
((b*sin(ang))+y)<<")"<<endl;
        }
    void eclipse:: rotate()
    { double angl;
     cout<< "enter angle to rotate the eclipse:";
     cin>>angl;
          ang+=angl;
    }
        void eclipse::print()
        {
        cout<<"Co-ordinates of center are ("<<x<<","<<y<<")"<<endl;
        cout<<"Major axis Radius is "<<a<<endl;
        cout<<"Minor axis Radius is "<<b<<endl;
        cout<<"Area of eclipse is "<<Area()<<endl;
        }
```

```
int main()
{
        circle objc;
        BaseConic* objp = NULL;
         objp = &objc;
        objc.print();
        objc.move(3,5);
        objc.print();
        objc.resize();
        objc.print();
        objc.position(3,5);
        objc.position(25,90);
        objc.position(4,3);

        eclipse obje;
        BaseConic* objep = NULL;
         objep = &obje;
        obje.print();
        obje.move(3,5);
        obje.print();
  obje.position();
        obje.resize();
        obje.print();
        obje.rotate();
        obje.print();
        obje.position();

        //BaseConic objcon;
         objp->print();
         objp->move(6,10);
```

```
        objp->print();
        objp->resize();
        objp->print();

        objep->print();
        objep->move(3,5);
       objep->print();
       objep->resize();
       objep->print();
       //objep->rotate();
       objep->print();

       return 0;
    }
```

PART B

```
#include<iostream>
#include<fstream>
#include<cstring>
#include<cstdlib>
#include<vector>

using namespace std;
int k;
class signal //Signal Class
{
   protected:
   int length;
   float max,aver;
   vector<float> Sdata;
   void average()
   {
```

```
    aver = 0 ;
    for(auto &i : Sdata)
    {
      aver += i ;
    }
    aver = aver/(float)length;
  }
  void maximum()
  {
    max = 0;
    for(auto &i : Sdata)
    {
      if(max < i )
        max = i;
    }
  }
  public:
friend signal operator+(const signal &ob1, const signal &ob2)throw(signal); // the function throws signal object as exception
friend void menu(signal &obj) throw(signal,int); // the function throws signal object,int as exception

  int sig_info()
  {
    //display sig_info
    maximum();
    average();
    cout<<endl<<endl<<"length\t\t:\t"<<length<<endl<<"maximum value\t:\t"<<max<<endl<<"average value\t:\t"<<aver;
    return 0;
  }
  int save_file() throw(string)
```

```
{
   //save code
   char save_file_name[20];
   cout<<"enter file name to save without extension";
   cin>>save_file_name;
                if(save_file_name == "Raw_data")
                        throw("wrong file name");
   strcat(save_file_name,".txt");
   maximum();
   fstream file;
   file.open(save_file_name,ios::out);
   file<<length<<" "<<max;
   for(auto i : Sdata)
   {
      file<<endl<<i;
   };
   file.close();
}
signal()
{
   length = 0;

}
signal(int number)
{
   char file_name[20]="Raw_data_";
   if(number&&number<10)
   {
      strcat(file_name,"0");
   }
   char file_number[5];
```

```
        sprintf(file_number,"%d",number);
        strcat(file_name,file_number);
        strcat(file_name,".txt");
        file_read(file_name);
    }
    signal(char* file_name)
    {
        file_read(file_name);
    }
    int file_read(char* file_name)
    {
        int un;
        float temp;
        fstream file;
        file.open(file_name,ios::in);
        file>>length>>un;
        for(int i=0;i<length;i++)
        {
            file>>temp;
            Sdata.push_back(temp);
            //cout<<signal_data[i]<<endl;
        }
        return 0;
    }
    ~signal()
    {
        //destructor
    }
    void operator+(float off)
{
```

```
        for(auto &i: Sdata)
        {
                i += off;
          cout<<i<<endl;
        }

}

 void operator*(int scale)
{
        for(auto &i: Sdata)
        {
                i *= scale;
          cout<<i<<endl;
        }
}

   void center(signal &y)
   {
      //center code
      average();
      y+( -aver );
   }
   void normalize(signal &y)
   {
      //normal code
      maximum();
      y*( (1 / float(max)) );

   }
   void statistics()
```

```
   {
      //statistics code
      maximum();
      average();
      sig_info();
   }

};

void menu(signal &obj) throw(signal,int)
{
   try{ // try block for the menu()
         menu1:
      cout<<endl<<endl<<"----------MENU----------
"<<endl<<"1.Offset"<<endl<<"2.Scale"<<endl<<"3.Center"<<endl<<"4.Normalize"<<endl<<"5.
Statistics"<<endl<<"6.Signal Information"<<endl<<"7.Save file"<<endl<<"8.
exit"<<endl<<endl<<"Enter your option  ";
      int choice=0;
      cin>>choice;
                switch(choice)
      {
         case 1 : try{
                        if(obj.length==0)
                              throw(obj);}
          catch(signal)
                        { cout<<"Empty object"<<endl;
                throw;
                        }
         float offset_value;
         cout<<"enter offset value ";
         cin>>offset_value;
         obj+(offset_value);
```

```
break;

case 2 : try{
                    if(obj.length==0)
                            throw(obj); }
    catch(signal)
                    { cout<<"Empty object"<<endl;
          throw;
                    }
float scale_factor;
cout<<"enter scale factor ";
cin>>scale_factor;
                    if(scale_factor==0)
                            throw(0);
                      obj*(scale_factor);
                    break;

case 3 :
obj.center(obj);
break;

case 4:
                    if(obj.length==0)
                            throw(obj);
obj.normalize(obj);
break;

case 5:
obj.statistics();
break;
```

```
        case 6:
        obj.sig_info();
        break;

        case 7:
                        try{
        obj.save_file();}
                        catch(string){
                                cout<<"file name exception"<<endl;}
        break;



        case 8: goto ret;
        break;

        default: cout<<"wrong choice"<<endl;
                        goto menul;
      }
goto menul;



}
       catch(int){ cout<<"Scaling by zero exception"<<endl; throw;} // catch an int exception
          catch(signal) // catches an signal object
          {
                  cout<<"Exception found empty object"<<endl;
                  throw;}
ret: cout<<"good bye"<<endl;
}
```

```
signal operator+(const signal &ob1, const signal &ob2) throw(signal)
{
        signal sum = ob1;
        try{ if(sum.length != ob2.length) //checks for exception in operator function
                throw(ob2);
        }
catch(signal)
{ cout<<"Operator exception"<<endl; throw; }
        if(sum.length == ob2.length)
    {
      for(int i=0; i < sum.length; i++ ){
       sum.Sdata[i] +=ob2.Sdata[i];
        cout<<sum.Sdata[i]<<endl;}
       k =1;
            }
     else
    { k=0;
     cout<<"addition is not possible"<<endl;
    }
 return sum;
}

int main(int argc,char* argv[])
{ try{ // check for exception in command line arguments
   if(argc==1)
        {
    try{
      signal ob;
      menu(ob);
        }
        catch(signal){ cout<<"caught exception"<<endl; throw;}
```

```
}
else
{
   if(argv[1][1]=='n')
   {
      signal ob(atoi(argv[2]));
      menu(ob);
   }
   else if(argv[1][1]=='f')
   {
      signal ob(argv[2]);
      menu(ob);
   }
   else if(argv[1][1]=='a')
   {
      signal ob1(atoi(argv[2]));
      signal ob2(atoi(argv[3]));
      signal ob3;
    try{
      ob3 = operator+(ob1, ob2);}

      catch(signal b){
                                    cout<<"signals are not of same length"<<endl;
                           throw(-1);}

      if ( k != 0)
      {
      ob3.statistics();
      menu(ob3);
      }
```

```
        }
        else
        {
           //input invalid
           cout<<"entered command line argument is wrong. please try again.";
throw('v');
        }

    }
}              catch(int){cout<<"signal not of same length exception"<<endl;}
catch(char){cout<<"command line argument exception"<<endl;}
                        catch(...){cout<<"exception found"<<endl;}
    return 0;
}
```

CONCLUSION

Thus learned usage of abstract base classes and exception handling