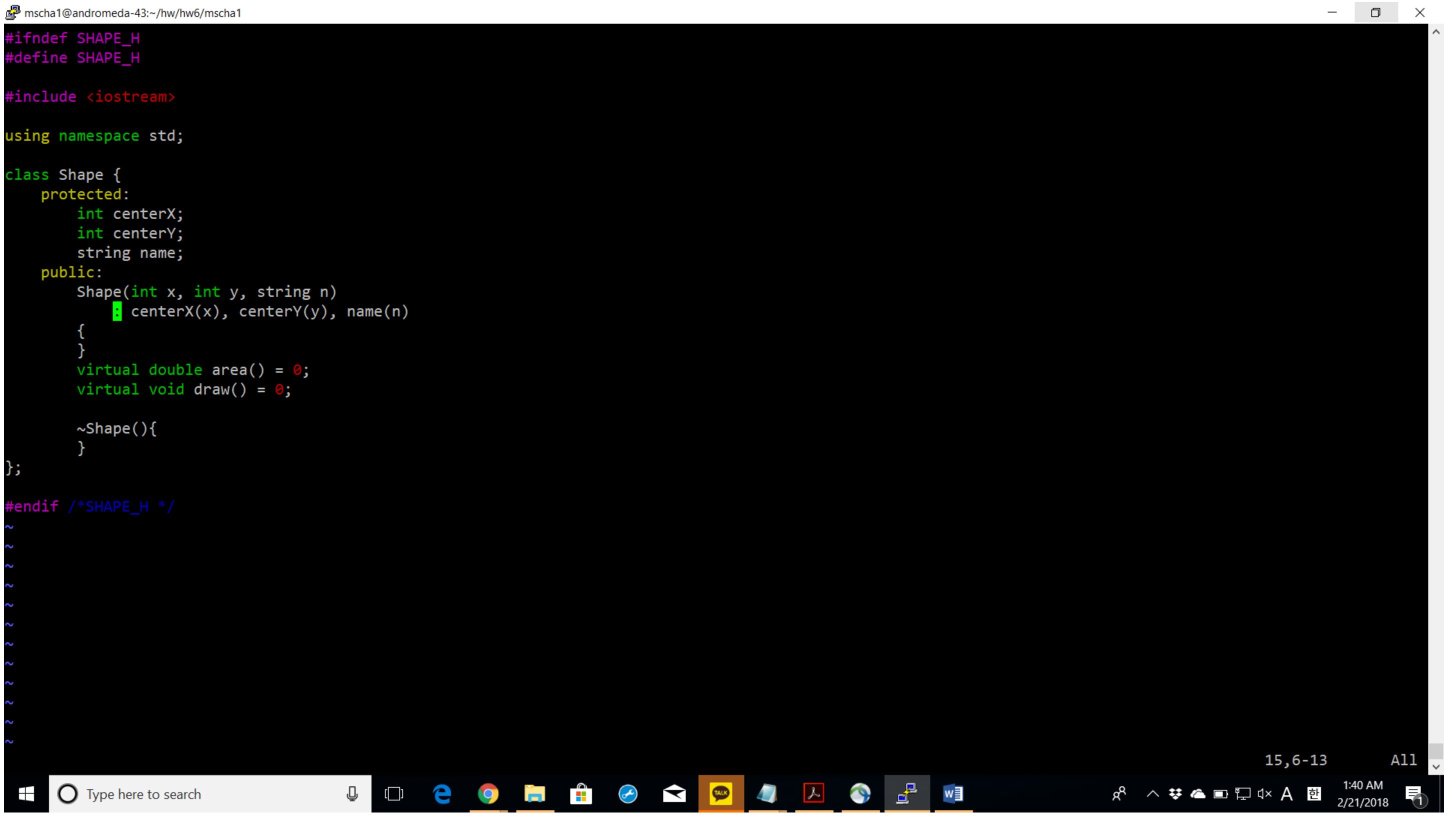
Name: Min Sung Cha Student ID: 85408485

Analysis of functions

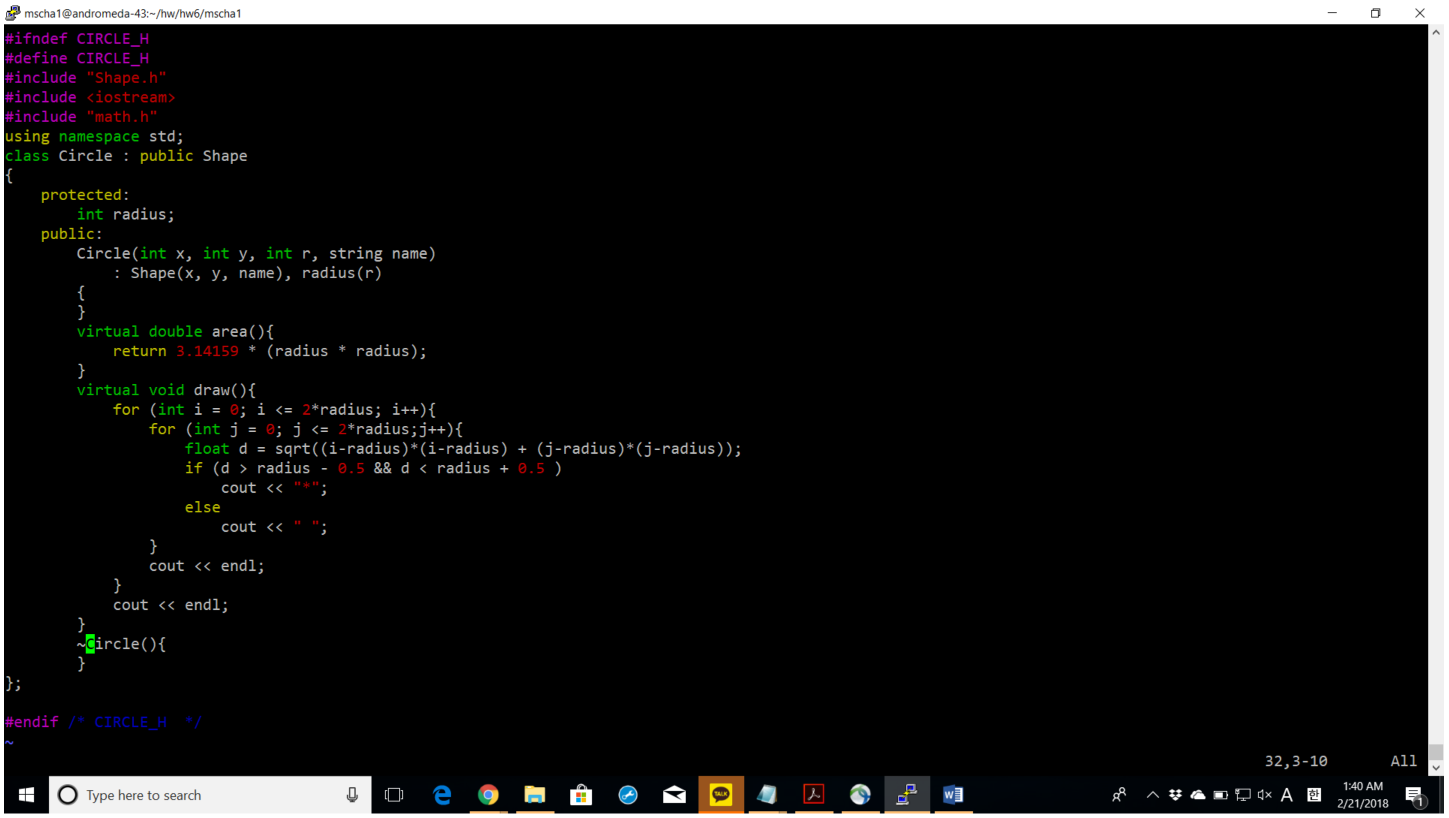
Shape.h



mscha1@andromeda-43:~/hw/hw6/mscha1

```
#ifndef SHAPE_H
#define SHAPE_H

#include <iostream>

using namespace std;

class Shape {
    protected:
        int centerX;
        int centerY;
        string name;
    public:
        Shape(int x, int y, string n)
            : centerX(x), centerY(y), name(n)
        {
        }
        virtual double area() = 0;
        virtual void draw() = 0;

        ~Shape(){
        }
};

#endif /*SHAPE_H */
~
~
~
~
~
~
~
~
~
~
~
~
~
                                                                15,6-13       All
```

Circle.h

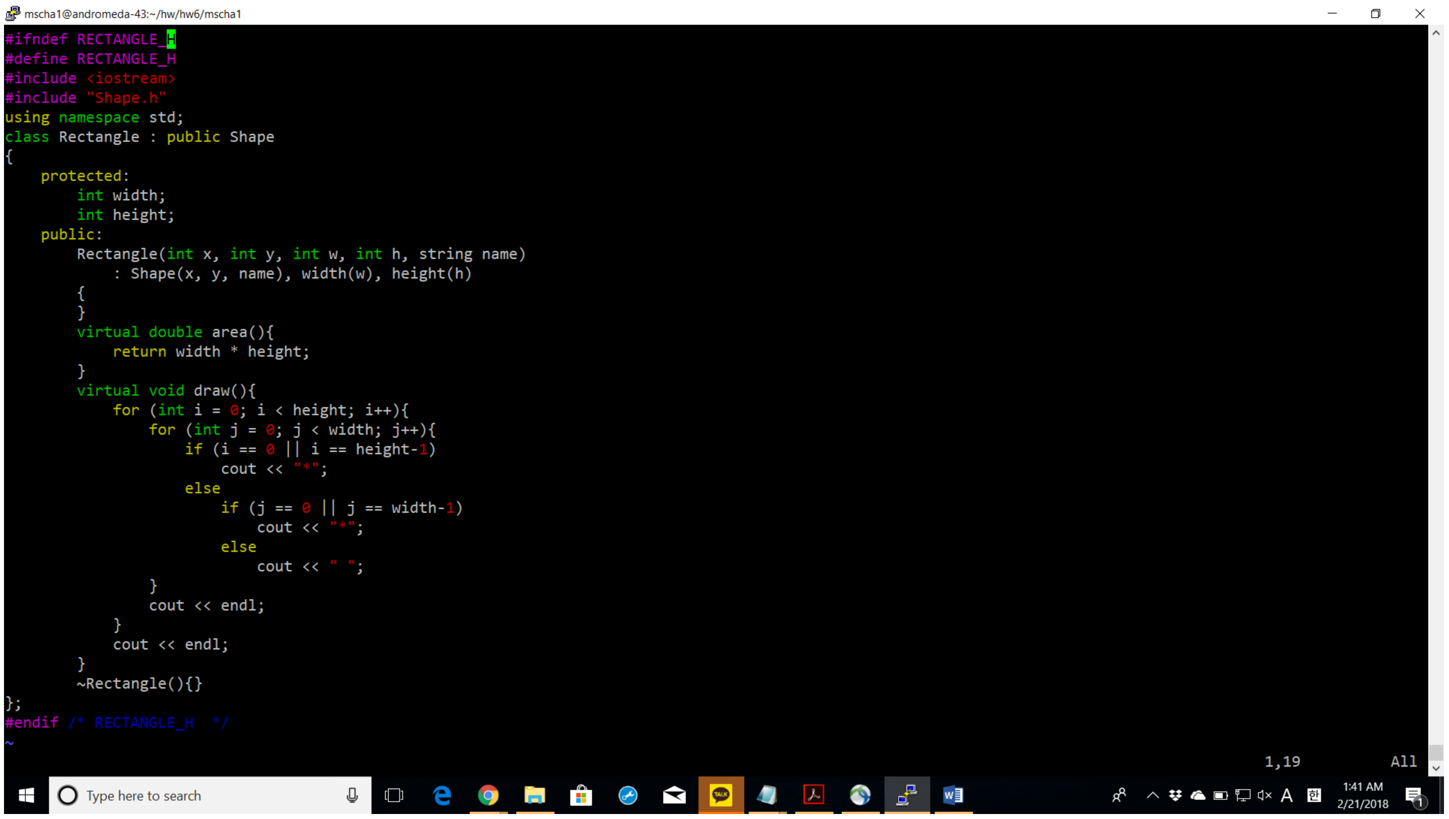

mscha1@andromeda-43:~/hw/hw6/mscha1

```
#ifndef CIRCLE_H
#define CIRCLE_H
#include "Shape.h"
#include <iostream>
#include "math.h"
using namespace std;
class Circle : public Shape
{
    protected:
        int radius;
    public:
        Circle(int x, int y, int r, string name)
            : Shape(x, y, name), radius(r)
        {
        }
        virtual double area(){
            return 3.14159 * (radius * radius);
        }
        virtual void draw(){
            for (int i = 0; i <= 2*radius; i++){
                for (int j = 0; j <= 2*radius;j++){
                    float d = sqrt((i-radius)*(i-radius) + (j-radius)*(j-radius));
                    if (d > radius - 0.5 && d < radius + 0.5 )
                        cout << "*";
                    else
                        cout << " ";
                }
                cout << endl;
            }
            cout << endl;
        }
        ~Circle(){
        }
};

#endif /* CIRCLE_H  */
~
                                                                     32,3-10       All
```

area: calculates the area by squaring the radius and multiplying it by pi.

Draw: calculates the distance from the center of the circle to the circumference and prints the * symbol accordingly.

Rectangle.h

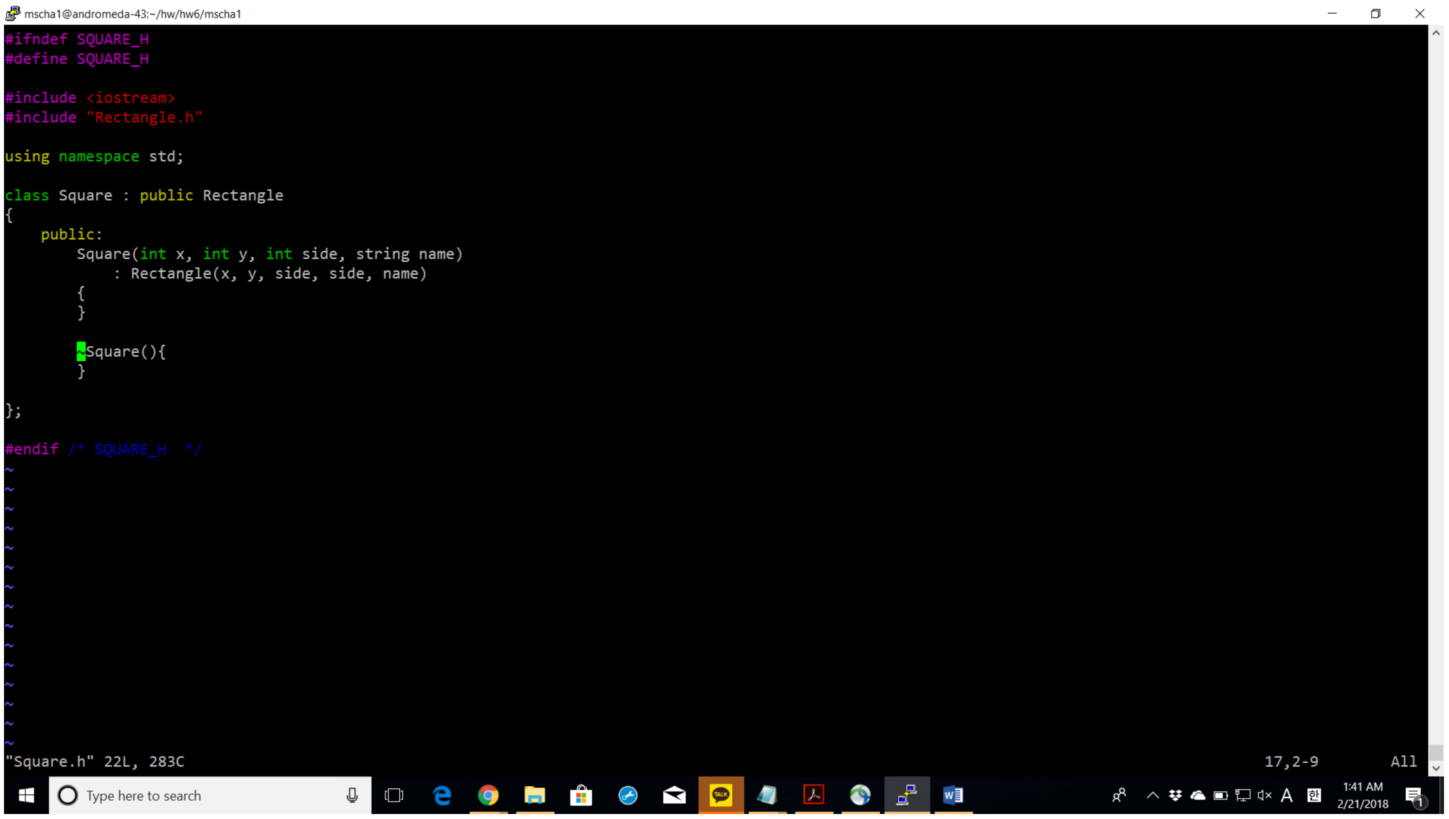


mscha1@andromeda-43:~/hw/hw6/mscha1

```
#ifndef RECTANGLE_H
#define RECTANGLE_H
#include <iostream>
#include "Shape.h"
using namespace std;
class Rectangle : public Shape
{
    protected:
        int width;
        int height;
    public:
        Rectangle(int x, int y, int w, int h, string name)
            : Shape(x, y, name), width(w), height(h)
        {
        }
        virtual double area(){
            return width * height;
        }
        virtual void draw(){
            for (int i = 0; i < height; i++){
                for (int j = 0; j < width; j++){
                    if (i == 0 || i == height-1)
                        cout << "*";
                    else
                        if (j == 0 || j == width-1)
                            cout << "*";
                        else
                            cout << " ";
                }
                cout << endl;
            }
            cout << endl;
        }
        ~Rectangle(){}
};
#endif /* RECTANGLE_H  */
~
                                                                                                                                              1,19          All
```

Draw: draws the entire line for the first and last row of the rectangle and for the rest of the rows only the two dots at the perimeter are drawn.

Square.h



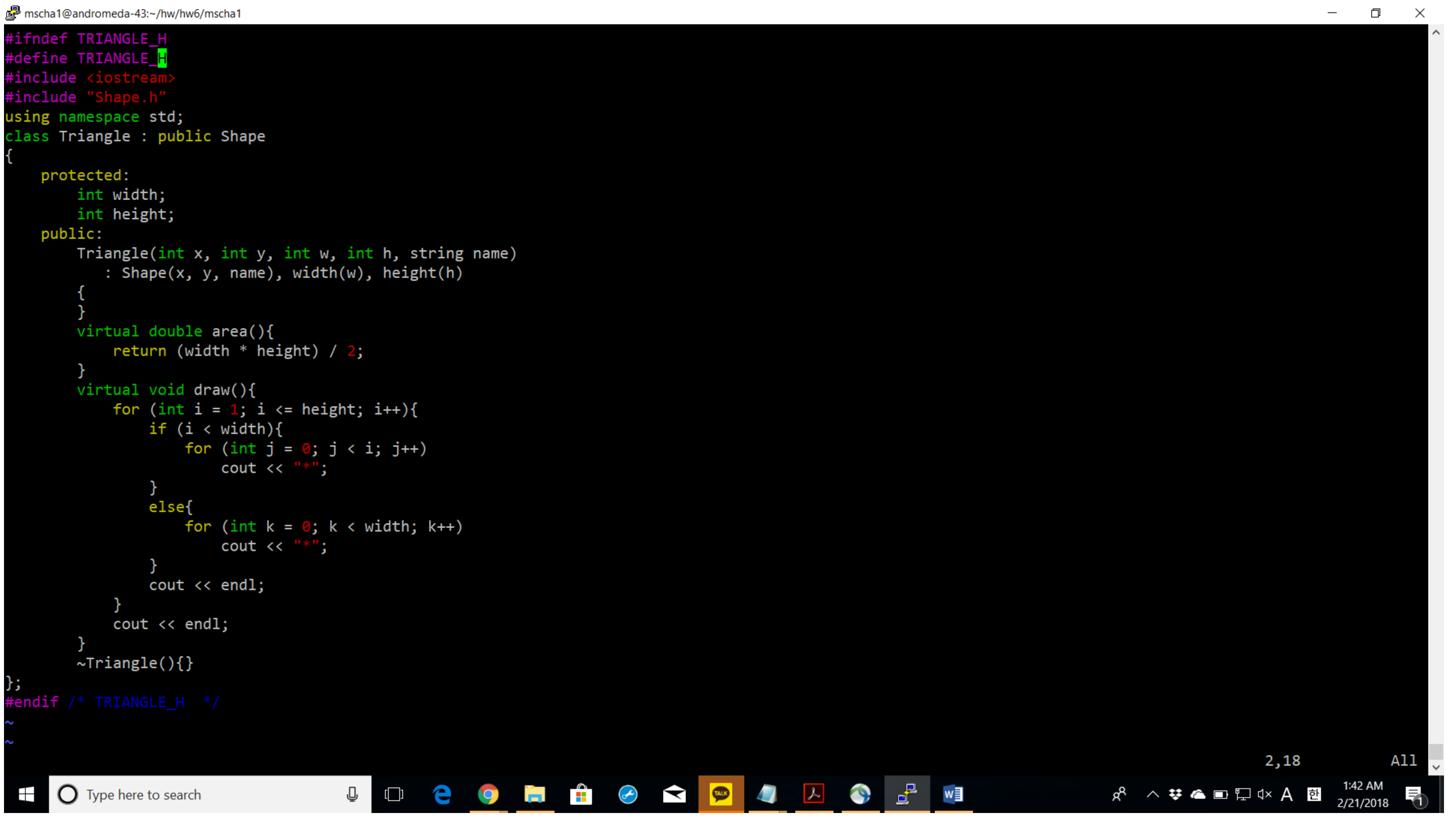
mscha1@andromeda-43:~/hw/hw6/mscha1

```
#ifndef SQUARE_H
#define SQUARE_H

#include <iostream>
#include "Rectangle.h"

using namespace std;

class Square : public Rectangle
{
    public:
        Square(int x, int y, int side, string name)
            : Rectangle(x, y, side, side, name)
        {
        }

        ~Square(){
        }

};

#endif /* SQUARE_H  */
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"Square.h" 22L, 283C                                                          17,2-9        All
```

Type here to search

1:41 AM 2/21/2018

Triangle.h



mscha1@andromeda-43:~/hw/hw6/mscha1

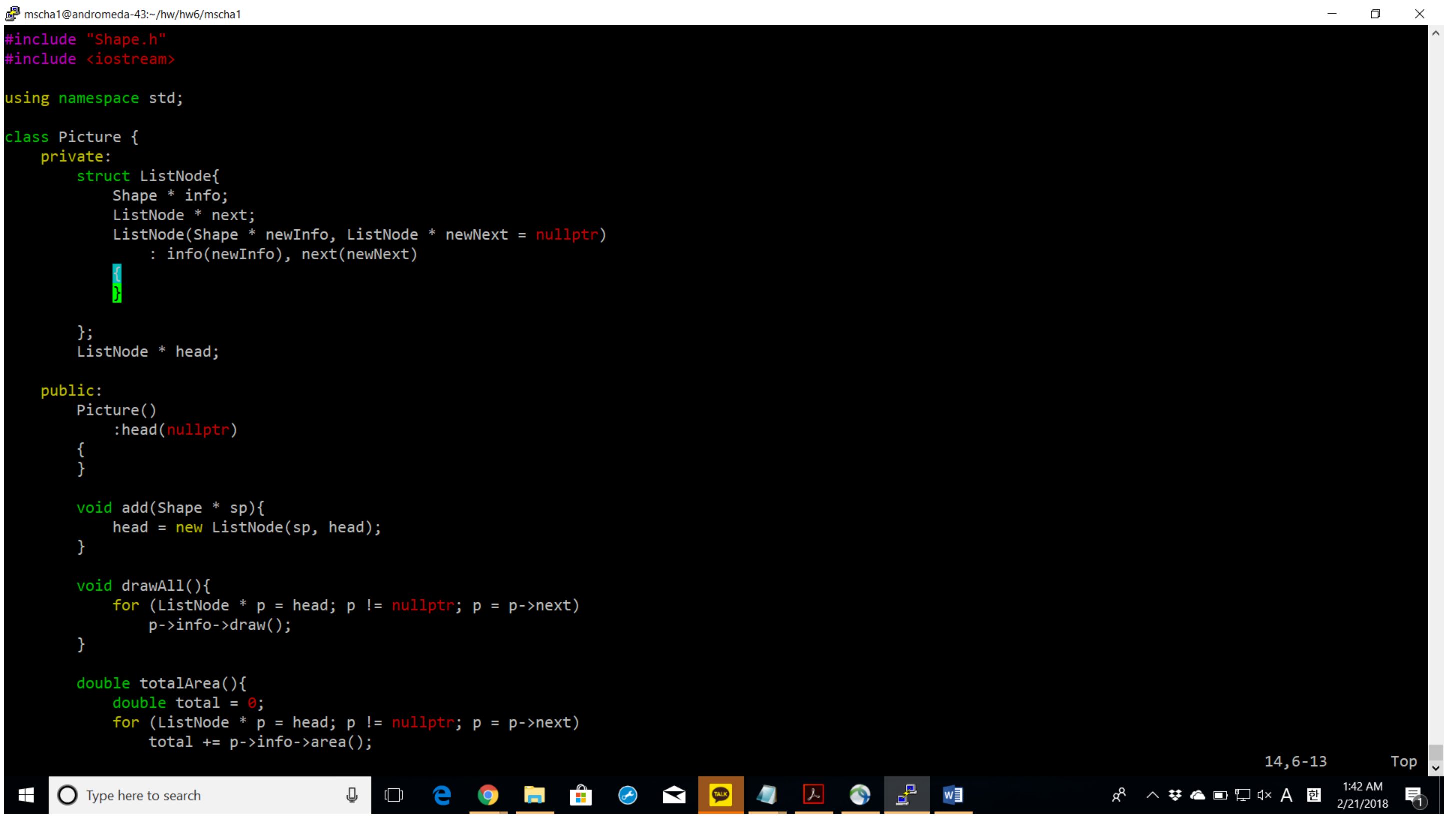
```
#ifndef TRIANGLE_H
#define TRIANGLE_H
#include <iostream>
#include "Shape.h"
using namespace std;
class Triangle : public Shape
{
    protected:
        int width;
        int height;
    public:
        Triangle(int x, int y, int w, int h, string name)
          : Shape(x, y, name), width(w), height(h)
        {
        }
        virtual double area(){
            return (width * height) / 2;
        }
        virtual void draw(){
            for (int i = 1; i <= height; i++){
                if (i < width){
                    for (int j = 0; j < i; j++)
                        cout << "*";
                }
                else{
                    for (int k = 0; k < width; k++)
                        cout << "*";
                }
                cout << endl;
            }
            cout << endl;
        }
        ~Triangle(){}
};
#endif /* TRIANGLE_H  */
~
~
                                                                              2,18          All
```

Type here to search

1:42 AM 2/21/2018

Draw: draws incrementally starting from the leftmost corner until the size of the base is reached. Once it is reached it keeps drawing the size of the base.

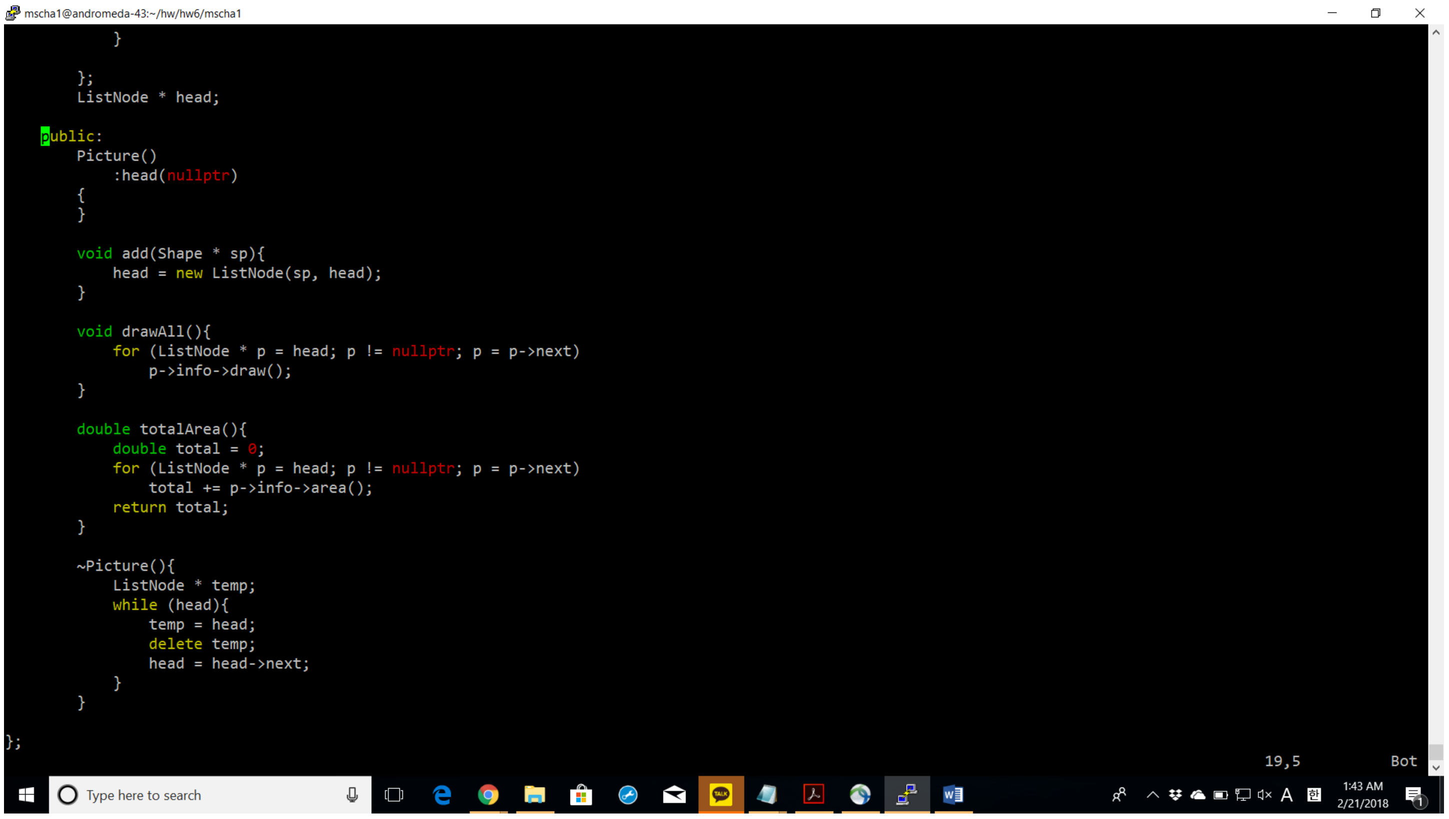
Picture.h



mscha1@andromeda-43:~/hw/hw6/mscha1

```
#include "Shape.h"
#include <iostream>

using namespace std;

class Picture {
    private:
        struct ListNode{
            Shape * info;
            ListNode * next;
            ListNode(Shape * newInfo, ListNode * newNext = nullptr)
                : info(newInfo), next(newNext)
            {
            }

        };
        ListNode * head;

    public:
        Picture()
            :head(nullptr)
        {
        }

        void add(Shape * sp){
            head = new ListNode(sp, head);
        }

        void drawAll(){
            for (ListNode * p = head; p != nullptr; p = p->next)
                p->info->draw();
        }

        double totalArea(){
            double total = 0;
            for (ListNode * p = head; p != nullptr; p = p->next)
                total += p->info->area();
                                                                                    14,6-13       Top
```

add: adds a list node at the beginning of the linked list.

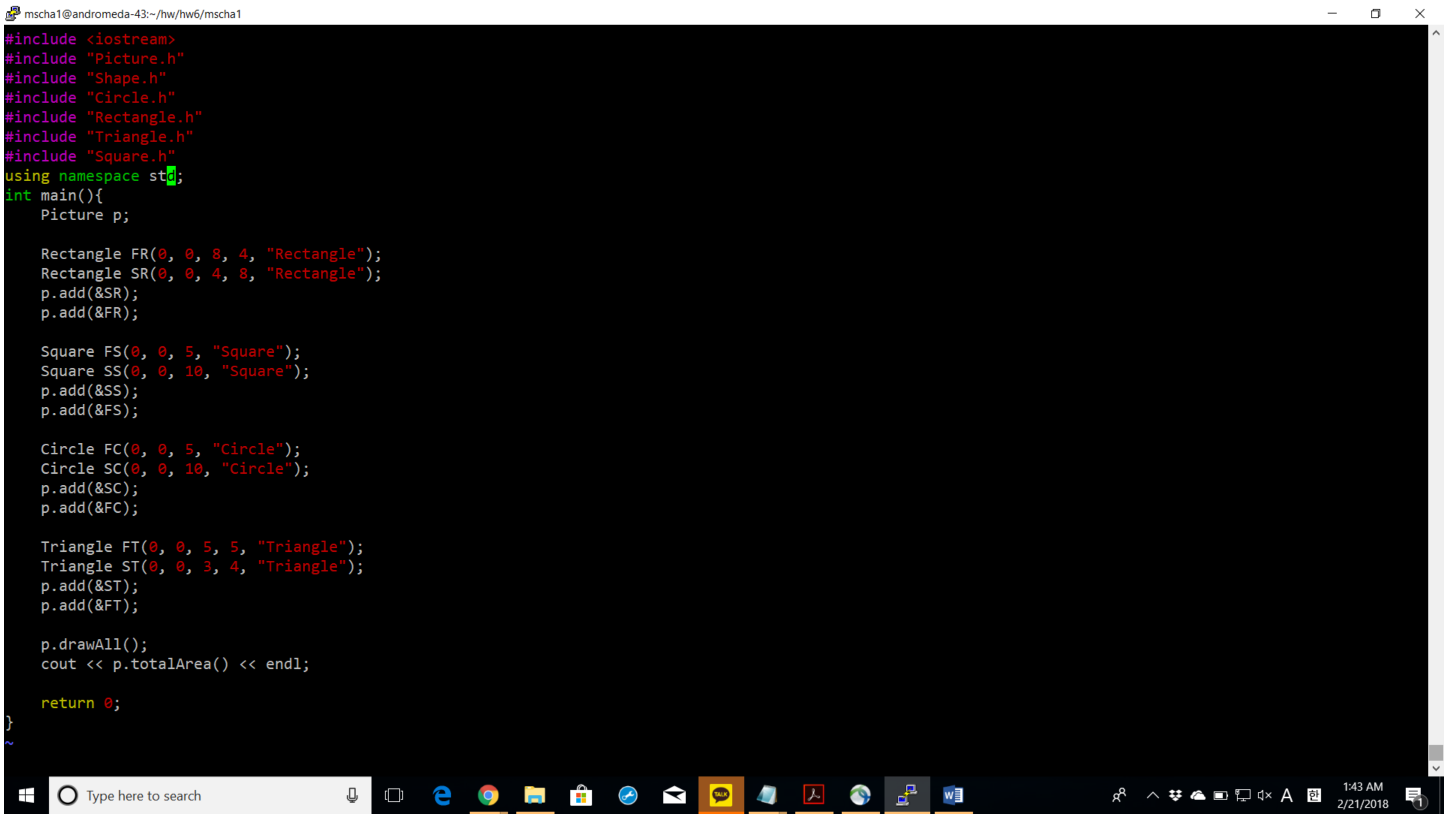


mscha1@andromeda-43:~/hw/hw6/mscha1

```
            }

        };
        ListNode * head;

    public:
        Picture()
            :head(nullptr)
        {
        }

        void add(Shape * sp){
            head = new ListNode(sp, head);
        }

        void drawAll(){
            for (ListNode * p = head; p != nullptr; p = p->next)
                p->info->draw();
        }

        double totalArea(){
            double total = 0;
            for (ListNode * p = head; p != nullptr; p = p->next)
                total += p->info->area();
            return total;
        }

        ~Picture(){
            ListNode * temp;
            while (head){
                temp = head;
                delete temp;
                head = head->next;
            }
        }

};
                                                                                    19,5          Bot
```

drawAll: draws all the Shape pointers inside the linked lists by referencing its info and then calling their virtual draw function.

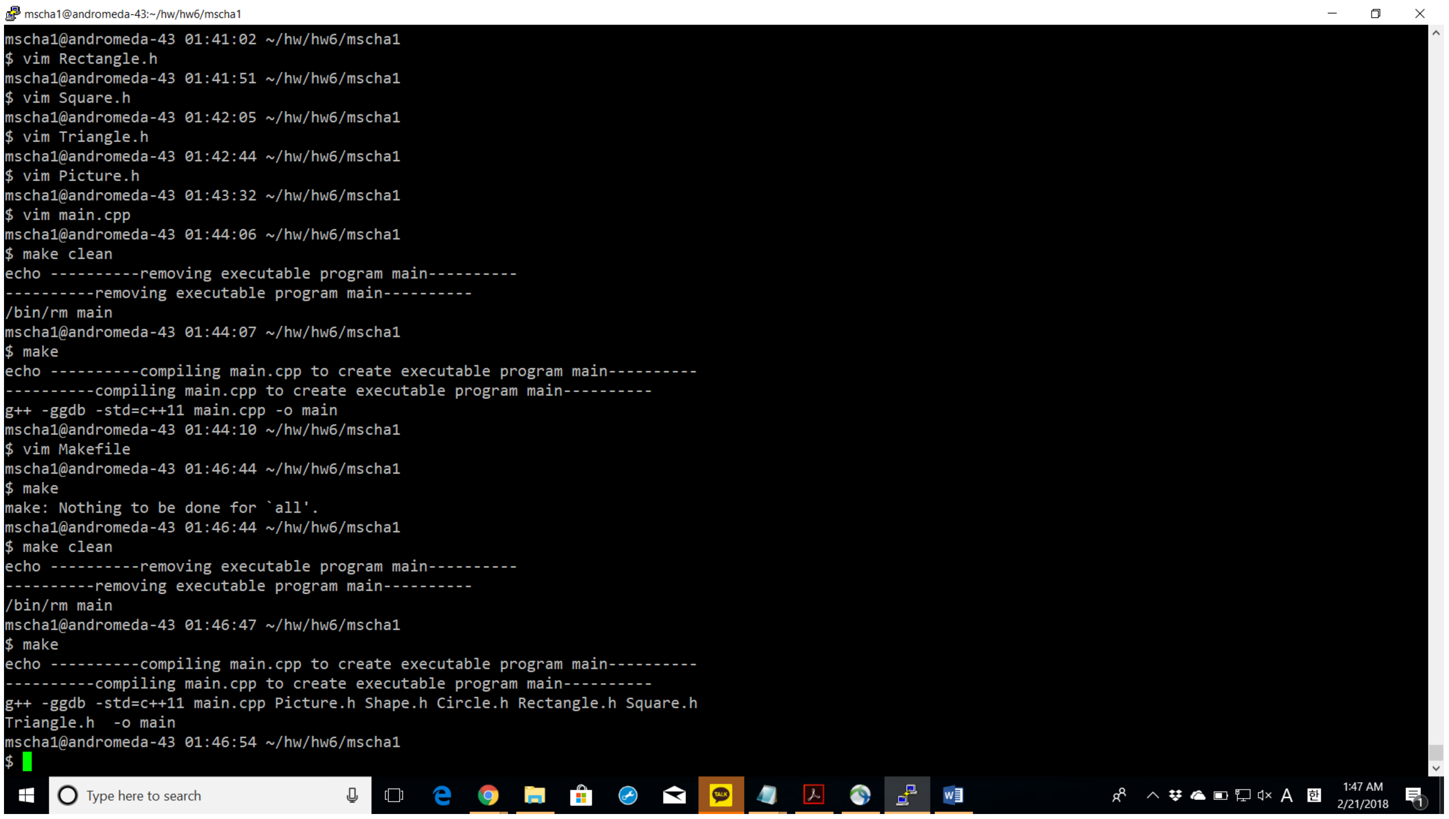
main.cpp



mscha1@andromeda-43:~/hw/hw6/mscha1

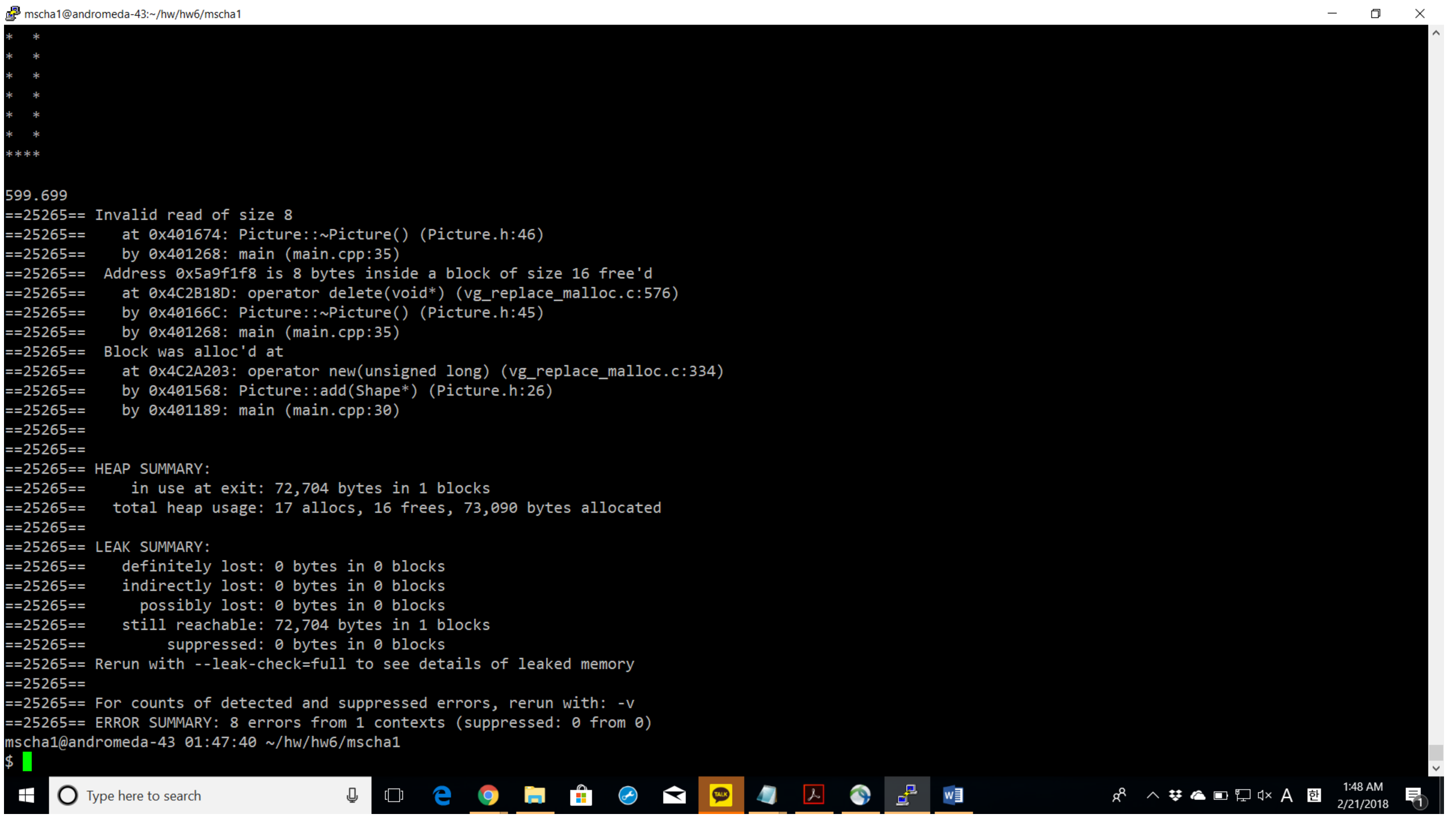
```
#include <iostream>
#include "Picture.h"
#include "Shape.h"
#include "Circle.h"
#include "Rectangle.h"
#include "Triangle.h"
#include "Square.h"
using namespace std;
int main(){
    Picture p;

    Rectangle FR(0, 0, 8, 4, "Rectangle");
    Rectangle SR(0, 0, 4, 8, "Rectangle");
    p.add(&SR);
    p.add(&FR);

    Square FS(0, 0, 5, "Square");
    Square SS(0, 0, 10, "Square");
    p.add(&SS);
    p.add(&FS);

    Circle FC(0, 0, 5, "Circle");
    Circle SC(0, 0, 10, "Circle");
    p.add(&SC);
    p.add(&FC);

    Triangle FT(0, 0, 5, 5, "Triangle");
    Triangle ST(0, 0, 3, 4, "Triangle");
    p.add(&ST);
    p.add(&FT);

    p.drawAll();
    cout << p.totalArea() << endl;

    return 0;
}
~
```

Screenshots of successful compiling with make command



```
mscha1@andromeda-43 01:41:02 ~/hw/hw6/mscha1
$ vim Rectangle.h
mscha1@andromeda-43 01:41:51 ~/hw/hw6/mscha1
$ vim Square.h
mscha1@andromeda-43 01:42:05 ~/hw/hw6/mscha1
$ vim Triangle.h
mscha1@andromeda-43 01:42:44 ~/hw/hw6/mscha1
$ vim Picture.h
mscha1@andromeda-43 01:43:32 ~/hw/hw6/mscha1
$ vim main.cpp
mscha1@andromeda-43 01:44:06 ~/hw/hw6/mscha1
$ make clean
echo ----------removing executable program main----------
----------removing executable program main----------
/bin/rm main
mscha1@andromeda-43 01:44:07 ~/hw/hw6/mscha1
$ make
echo ----------compiling main.cpp to create executable program main----------
----------compiling main.cpp to create executable program main----------
g++ -ggdb -std=c++11 main.cpp -o main
mscha1@andromeda-43 01:44:10 ~/hw/hw6/mscha1
$ vim Makefile
mscha1@andromeda-43 01:46:44 ~/hw/hw6/mscha1
$ make
make: Nothing to be done for `all'.
mscha1@andromeda-43 01:46:44 ~/hw/hw6/mscha1
$ make clean
echo ----------removing executable program main----------
----------removing executable program main----------
/bin/rm main
mscha1@andromeda-43 01:46:47 ~/hw/hw6/mscha1
$ make
echo ----------compiling main.cpp to create executable program main----------
----------compiling main.cpp to create executable program main----------
g++ -ggdb -std=c++11 main.cpp Picture.h Shape.h Circle.h Rectangle.h Square.h
Triangle.h  -o main
mscha1@andromeda-43 01:46:54 ~/hw/hw6/mscha1
$ 
```

Screenshots of successful execution with valgrind command.



mscha1@andromeda-43:~/hw/hw6/mscha1

```
*  *
*  *
*  *
*  *
*  *
*  *
****


599.699
==25265== Invalid read of size 8
==25265==    at 0x401674: Picture::~Picture() (Picture.h:46)
==25265==    by 0x401268: main (main.cpp:35)
==25265==  Address 0x5a9f1f8 is 8 bytes inside a block of size 16 free'd
==25265==    at 0x4C2B18D: operator delete(void*) (vg_replace_malloc.c:576)
==25265==    by 0x40166C: Picture::~Picture() (Picture.h:45)
==25265==    by 0x401268: main (main.cpp:35)
==25265==  Block was alloc'd at
==25265==    at 0x4C2A203: operator new(unsigned long) (vg_replace_malloc.c:334)
==25265==    by 0x401568: Picture::add(Shape*) (Picture.h:26)
==25265==    by 0x401189: main (main.cpp:30)
==25265==
==25265==
==25265== HEAP SUMMARY:
==25265==     in use at exit: 72,704 bytes in 1 blocks
==25265==   total heap usage: 17 allocs, 16 frees, 73,090 bytes allocated
==25265==
==25265== LEAK SUMMARY:
==25265==    definitely lost: 0 bytes in 0 blocks
==25265==    indirectly lost: 0 bytes in 0 blocks
==25265==      possibly lost: 0 bytes in 0 blocks
==25265==    still reachable: 72,704 bytes in 1 blocks
==25265==         suppressed: 0 bytes in 0 blocks
==25265== Rerun with --leak-check=full to see details of leaked memory
==25265==
==25265== For counts of detected and suppressed errors, rerun with: -v
==25265== ERROR SUMMARY: 8 errors from 1 contexts (suppressed: 0 from 0)
mscha1@andromeda-43 01:47:40 ~/hw/hw6/mscha1
$
```