Assignment 1

Take a look at this example code:

```
#include <stdio.h>

class Shape {
public:
        virtual ~Shape();
        virtual void draw() = 0;
};

class Circle : public Shape {
public:
        virtual ~Circle();
        virtual void draw();
};

Shape::~Shape() {
        printf("shape destructor\n");
}

// void Shape::draw() {
//      printf("Shape::draw\n");
// }

Circle::~Circle() {
        printf("circle destructor\n");
}

void Circle::draw() {
        printf("Circle::draw\n");
}


int main() {
        Shape *shape = new Circle;
        shape->draw();
        delete shape;

        return 0;
}
```

Verify your understanding of how the `virtual` keyword and method overriding work by performing a few experiments:

1. Remove the `virtual` keyword from each location individually, recompiling and running each time to see how the output changes. Can you predict what will and will not work?
2. Try making `Shape::draw` non-pure by removing `= 0` from its declaration.
3. Try changing `shape` (in `main()`) from a pointer to a stack-allocated variable.