Question.1

```
#include<iostream>
#include<math.h>

using namespace std;

class Triangle
{
	public:
	void area(int a,int b,int c);
	void perimeter(int a,int b,int c);
};

void Triangle::area(int a,int b,int c)
{
	float s=((a+b+c)/2.0),A=sqrt(s*(s-a)*(s-b)*(s-c));
	cout<<"Area of a triangle is: "<<A<<" sq. units\n";
}

void Triangle::perimeter(int a,int b,int c)
{
	cout<<"Perimeter of a triangle is: "<<a+b+c<<" units\n";
}

int main()
{
	Triangle tr1;
	int side1=3,side2=4,side3=5;
```

```
        tr1.area(side1,side2,side3);
        tr1.perimeter(side1,side2,side3);
        return 0;
}
```

Question.2

```
#include <iostream>

using namespace std;

class Rect
{
private:
   int a, b;

public:
   Rect(int a, int b)
   {
      this->a = a;
      this->b = b;
   }
   int area()
   {
      return this->a * this->b;
   }
};

int main()
```

```
{
    Rect obj = Rect(3, 4);
    cout << obj.area() << "\n";

    return 0;
}
```

Question.3

```
#include<iostream>
using namespace std;
class Area{
        public:
                int len,width;
                void getArea(){
                        cout<<"Enter the length: ";
                        cin>>len;
                        cout<<"Enter width: ";
                        cin>>width;
                }
                void returnArea(){
                        cout<<"Area is: "<<len*width;
                }
};
int main(){
        Area A;
        A.getArea();
        A.returnArea();
}
```

Question.4

```
#include <iostream>
using namespace std;
class Average{
   public:
   static float calcAverate(float a, float b, float c){
      return (a + b + c) / 3;
   }
};


int main(){
   cout<<"Enter three numbers: ";
   float a, b, c;
   cin>>a;
   cin>>b;
   cin>>c;
   cout<<"The average is: "<<Average::calcAverate(a,b,c)<<endl;
   return 0;
}
```

Question.5

```
#include<iostream>

using namespace std;

class Complex
```

```
{
        double re;
        double im;
public:
        //Default constructor
        Complex(){}
        //Init constructor
        Complex(double _re, double _im)
        :re(_re),im(_im){}
        //Copy constructor
        Complex(Complex& x)
        {
                re=x.re;
                im=x.im;
        }
        Complex operator+ (Complex& x)
        {
                re=re+x.re;
                im=im+x.im;
                return *this;
        }
        Complex operator- (Complex& x)
        {
                re=re-x.re;
                im=im-x.im;
                return *this;
        }
        Complex operator* (Complex& x)
        {
```

```
                re=re*x.re-im*x.im;
                im=re*x.im+x.re*im;
                return *this;
        }
        friend ostream& operator<<(ostream&, Complex&);
        friend istream& operator>>(istream&, Complex&);
//      friend Complex operator+(Complex&,Complex&);
};
istream& operator>> (istream& is, Complex& x)
{
        cout<<"Please, enter a real part of complex number: ";
        is>>x.re;
        cout<<"Please, enter an imaginary part of complex number: ";
        is>>x.im;
        return is;
}

ostream& operator<< (ostream& os, Complex& x)
{
        os<<x.re;
        if(x.im>0)
        {
                os<<"+"<<x.im<<"i";
        }
        else if(x.im<0)
        {
                os<<x.im<<"i";
        }
        return os;
```

```
}



int main()
{
	Complex a,b;
	cin>>a;
	cin>>b;
	cout<<"You entered two complex numbers:\n";
	cout<<"a= "<<a
		<<"\nb= "<<b<<endl;
	a+b;
	cout<<"a+b= "<<a<<endl;
	a-b;
	cout<<"a-b= "<<a<<endl;
	a*b;
	cout<<"a*b= "<<a<<endl;
}
```