Dry part

Question one:

a. There are 3 mistakes:

1. **Line 13:** out << "B: " << n;
   Simply using n will not work, as there is no B object that calls this function, but it is rather an ostream.
   Therefore, the function must use the reference to the B object in order to retrieve n.
   Correction: out << "B: " << b.n;

2. **Line 22:** if(b1 < b2)
   b1 is a const, while the function is not. Therefore it assums that it will change the contents of the object.
   For the fix, the operater overload function was changed to be a const function.
   Correction: bool operator<(const B &rhs) const

3. **Line 30:** const B b4 = b1 + (b2 + b3);
   Operator overloads cant take temporary values. Therefore, we have to start a new variable.
   Correction: B b_temp = b2 + b3; const B b4 = b1 + b_temp;

b. <u>what does this code output?</u>

Chain of calls:

- *Defining "pa" calls for the constructor of A.*
  Which is silent. No output
  *Then prints: "**applying function f:**"*
- *Now function "f" is called from main, which takes "pa" and copies is to "a" and that calls the copy constructor of A, and prints: "**A copy ctor**".*
- *From "f" a.type() is called and it prints: "**This is A**".*
- *Type() returns a copy of "a" and that calls for the copy constructor of A. which prints: "**A copy ctor**".*
- *And once more type is called ad prints: "**this is A**".*
- *Now the two copies from before should be deleted, the A destructor is called, and it prints: "**A dtor**" .*
- *Next line is: "**applying function g**"*
- *Now the function "g" is called but it does not use the copy constructor of A because it takes a reference.*
- *a.type is called from "g" and prints: "**this is B**".*
  *and since a.type returns a reference the copy constructor is not called and now .type is called again from main and prints: "**this is B**" because "pa" is B.*
- *now what is left is for the destructor of B and A to do their job, so prints: "**B dtor**" and "**A dtor**".*

So the output should be:

1. applying function f:
2. A copy ctor
3. This is A
4. A copy ctor
5. This is A
6. A dtor
7. applying function g:
8. This is B
9. This is B
10. B dtor
11. A dtor

Question two:

a. The “Car” class:

```
class Car {
public:

     virtual int getFuelConsumption(int speed) const=0;

     Car() = default;
     ~Car() = default;
};
```

b. getPetrol function:

```
double getPetrol(std::vector<Road> roads, const Car& car) {
double petrol=0;
for (vector<Road>::iterator i = roads.begin(); i != roads.end();
++i) {
      petrol += (*i).length*car.getFuelConsumption((*i).speed);
      }
      return petrol;
}
```