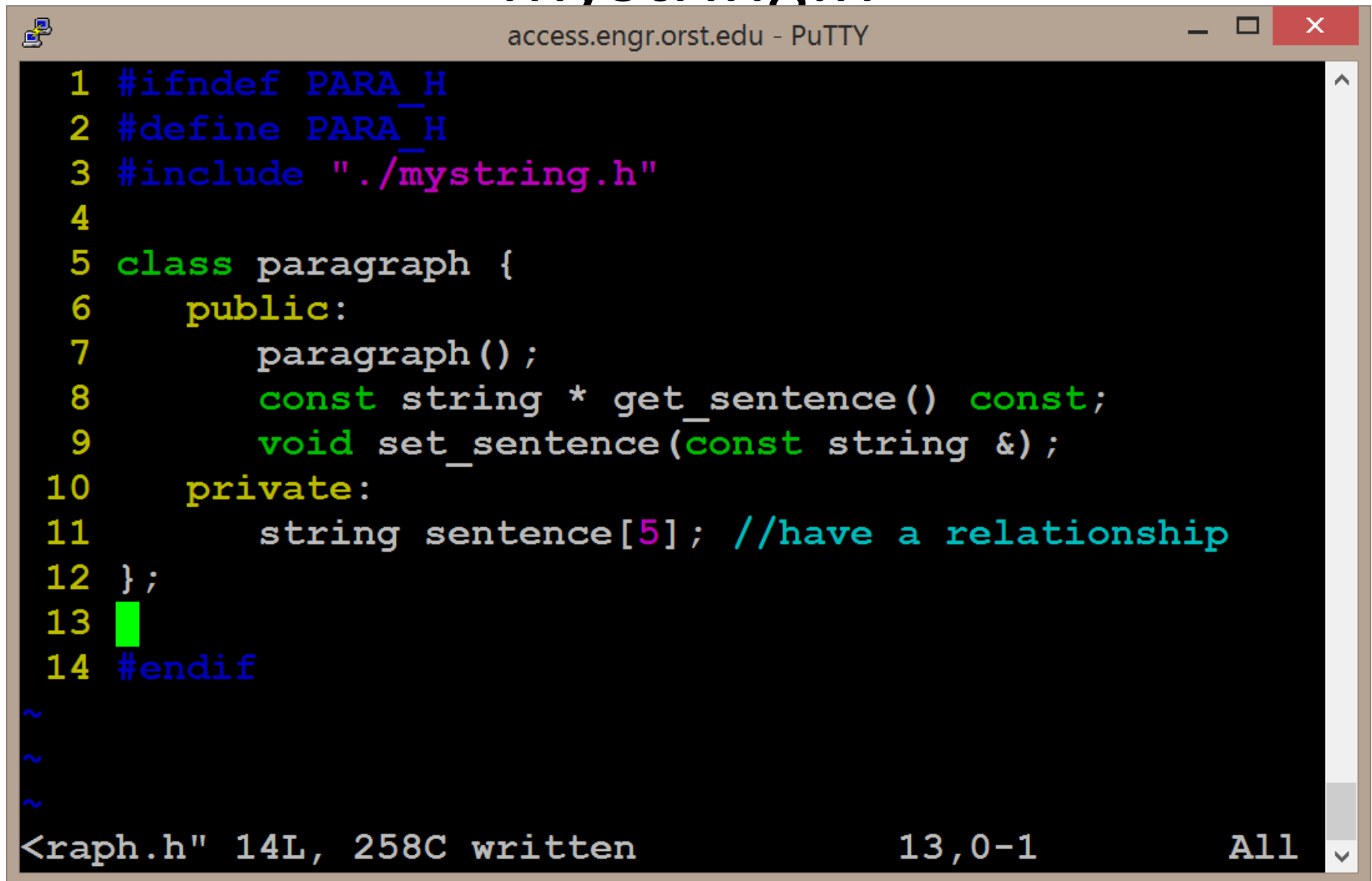


CS 162

Intro to CS II

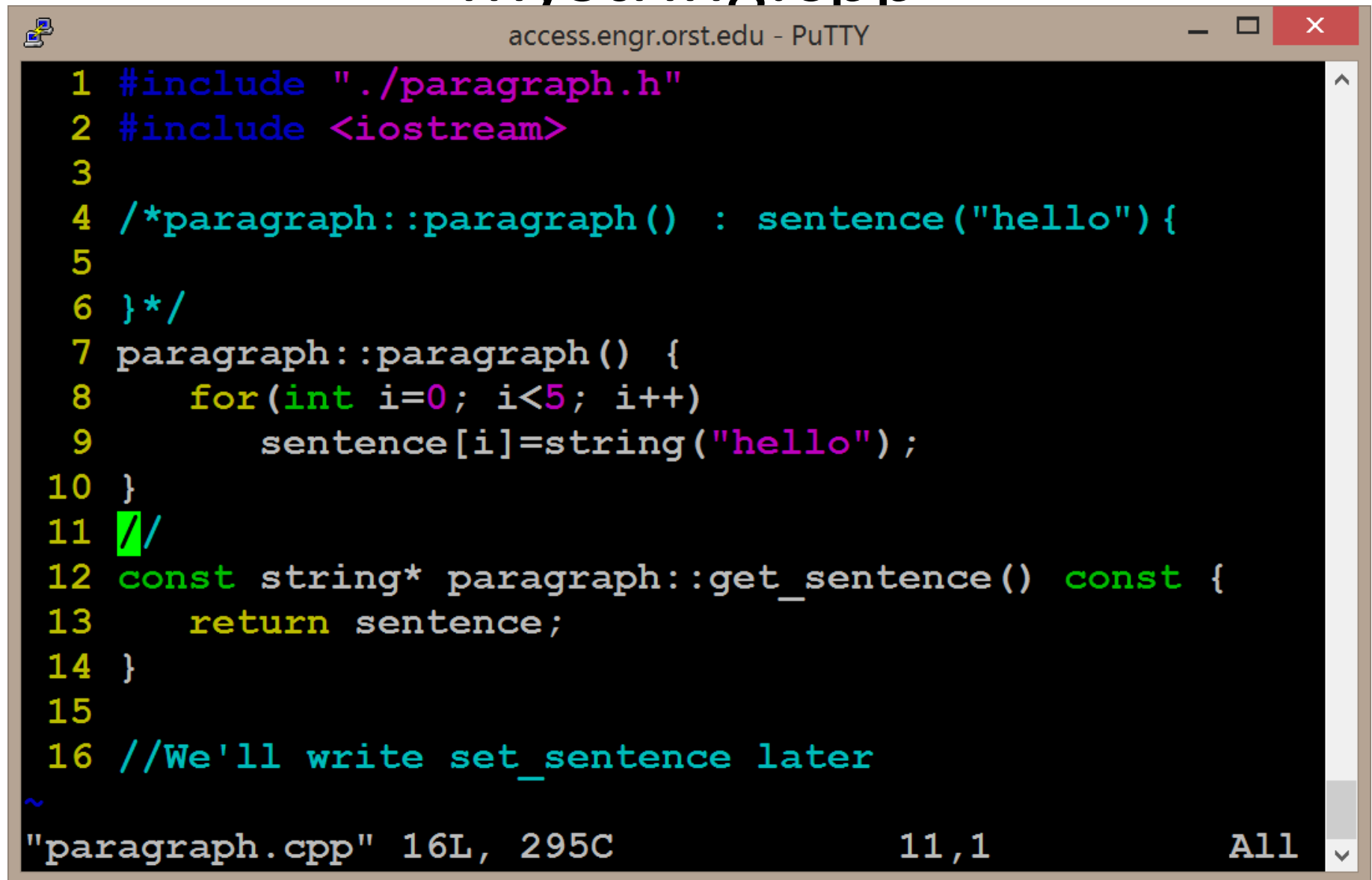
“Has a” vs. “Is a” Relationship

mystring.h



```
1 #ifndef PARA_H
2 #define PARA_H
3 #include "../mystring.h"
4
5 class paragraph {
6     public:
7         paragraph();
8         const string * get_sentence() const;
9         void set_sentence(const string &);
10    private:
11        string sentence[5]; //have a relationship
12 };
13
14 #endif
~
~
~
<raph.h" 14L, 258C written      13,0-1      All
```

mystring.cpp



```
1 #include "../paragraph.h"
2 #include <iostream>
3
4 /*paragraph::paragraph() : sentence("hello"){
5
6 }*/
7 paragraph::paragraph() {
8     for(int i=0; i<5; i++)
9         sentence[i]=string("hello");
10 }
11 //
12 const string* paragraph::get_sentence() const {
13     return sentence;
14 }
15
16 //We'll write set_sentence later
~
"paragraph.cpp" 16L, 295C 11,1 All
```

main.cpp

```
1 #include <iostream>
2 #include "../mystring.h"
3 #include <fstream>
4 #include <stdio.h>
5 #include "../paragraph.h"
6 using std::cout;
7 using std::endl;
8 using std::fstream;
9 using std::ios;
10
11 int main() {
12     paragraph p;
13     cout << p.get_sentence()[2].at(1) << endl;
14
15     string str2("hello");
16     //string str=str2; //supposed to call copy const
    ruct to make new
"main.cpp" 40L, 823C 14,3 Top
```

What is inheritance?

- Webster Definition?
 - the reception of genetic qualities by transmission from parent to offspring
 - the acquisition of a possession, condition, or trait from past generations
- CS Definition?
 - Base class (Parent) and Derived class (Child)
 - Ancestor class and Descendant class (generations)

Superclass

Subclass

Vehicle + Bike Inheritance Interface

not accessible by child

```
class parent {  
    public:  
        parent(); //Have a constructor  
        void print_mssg();  
        int get_shared_var();  
    private:  
        int shared_var;  
};
```

this is inherited by child

inherits from "is a"

```
class child : public parent {  
    public:  
        child(); //This constructor needs to call parent() constructor  
        void print_mssg(); //Redefine or Override inherited function  
    private:  
        int unique_var;  
};
```

** Your child can directly access, protected*

Inheritance Implementation

```
parent::parent() {  
    shared_var = 0;  
}  
int parent::get_shared_var() {  
    return shared_var;  
}  
void parent::print_mssg() {  
    cout << "I'm parent!" << endl;  
}
```

//child class implementation

```
child::child() : parent() { //Need to call inherited constructor first  
    unique_var = 0;  
}  
void child::print_mssg() {  
    cout << "I'm child!" << endl; //This will take precedence over parent  
}
```

What is not inherited?

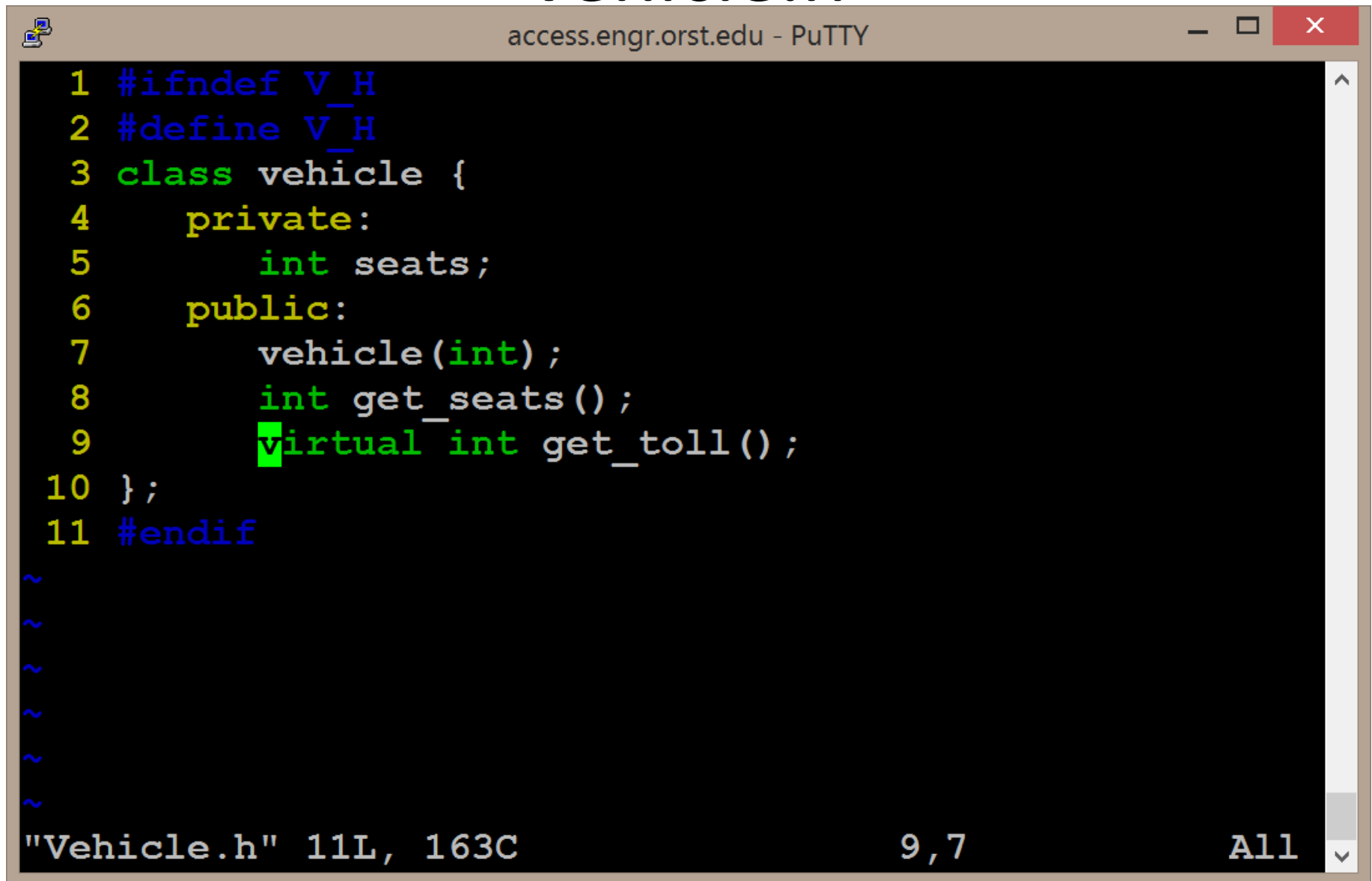
*implicitly called
vs. inherited.*

- Constructors
- Destructors
- Friends
- Assignment Op Overload
- Inherited, but not accessible: Private Members

Demo: Vehicle Toll ...

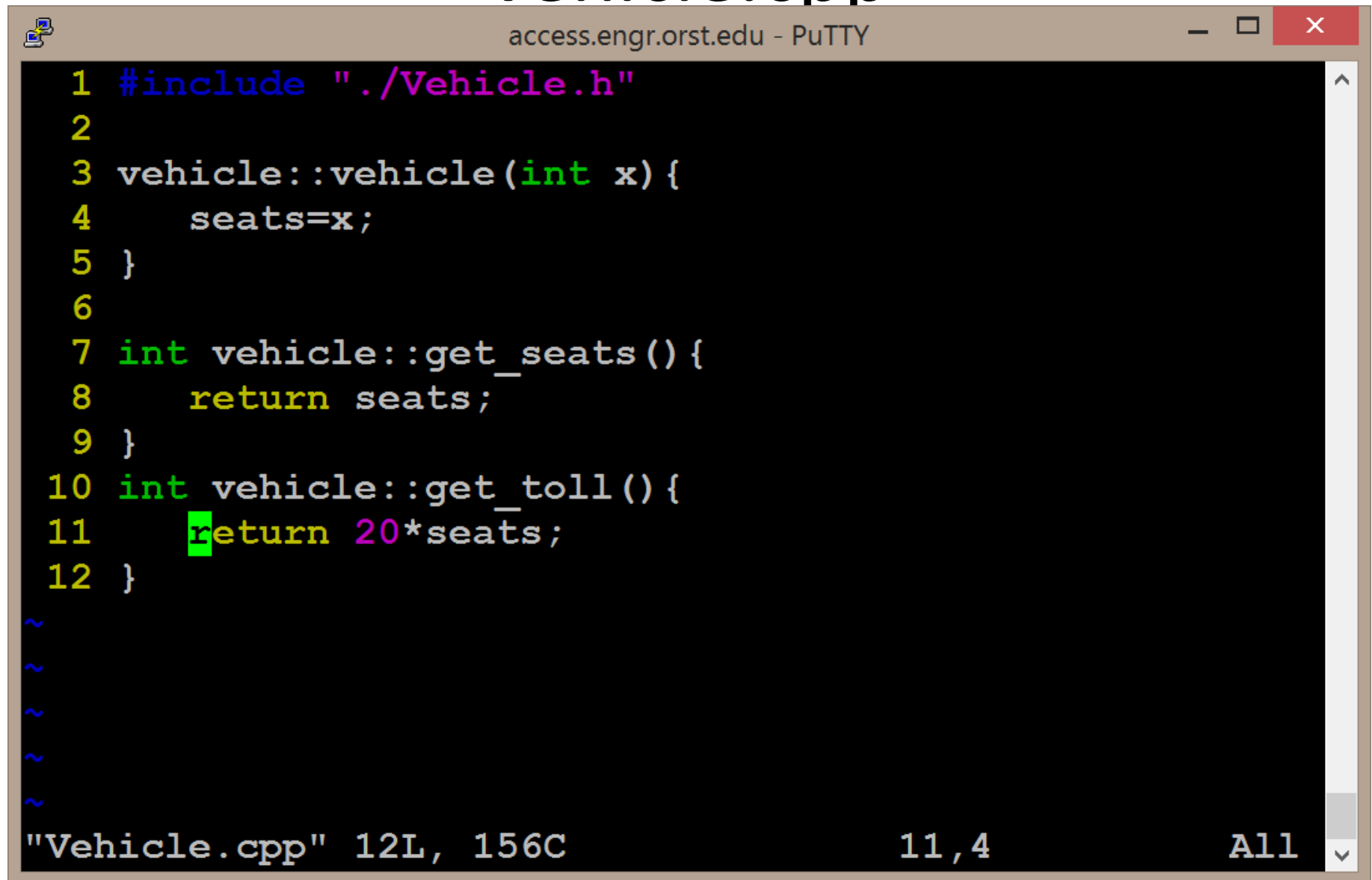
- Get into groups of 4-5 people
- Design the classes for a vehicle and bike to provide the toll amount based on the seats for all vehicles, except bikes that are free.
 - Non-default constructors to set the seats
 - Accessor function for the seats
 - Provide toll amount for vehicles and bikes
- How will you make sure it is working?

vehicle.h



```
1 #ifndef V_H
2 #define V_H
3 class vehicle {
4     private:
5         int seats;
6     public:
7         vehicle(int);
8         int get_seats();
9         virtual int get_toll();
10 };
11 #endif
~
~
~
~
~
~
~
"Vehicle.h" 11L, 163C          9,7          All
```

vehicle.cpp



```
1 #include "../Vehicle.h"
2
3 vehicle::vehicle(int x) {
4     seats=x;
5 }
6
7 int vehicle::get_seats() {
8     return seats;
9 }
10 int vehicle::get_toll() {
11     return 20*seats;
12 }
~
~
~
~
~
~
"Vehicle.cpp" 12L, 156C 11,4 All
```

bike.h

```
1 #ifndef B_H
2 #define B_H
3 #include "../Vehicle.h"
4 class bike : public vehicle {
5     public:
6         bike(int) ;
7         int get_toll() ;
8 };
9 #endif
```

~~~~~

```
"bike.h" 9L, 137C
```

4,1

All

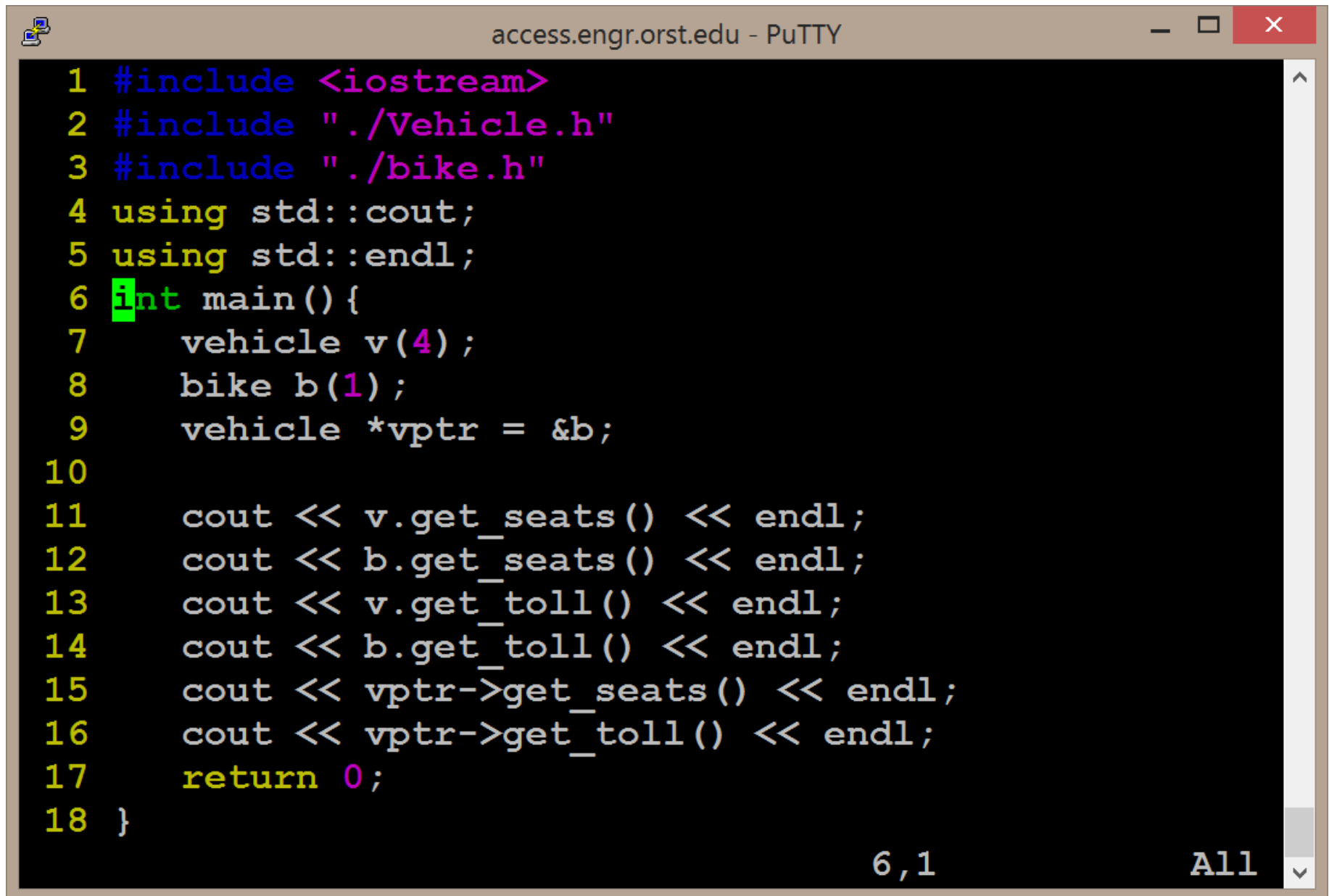
# bike.cpp

A screenshot of a PuTTY terminal window titled "access.engr.orst.edu - PuTTY". The terminal displays C++ code for a file named "bike.cpp". The code consists of seven lines:  

```
1 #include "../bike.h"  
2  
3 bike::bike(int x) : vehicle(x) {}  
4  
5 int bike::get_toll() {  
6     return 0;  
7 }  
~  
~  
~  
~  
~  
~  
~  
~  
~
```

  
The status bar at the bottom shows the filename and dimensions as "`bike.cpp`" 7L, 90C. On the right side of the status bar, it indicates the cursor position as 6,4 and the selection mode as All.

# main.cpp



```
1 #include <iostream>
2 #include "../Vehicle.h"
3 #include "../bike.h"
4 using std::cout;
5 using std::endl;
6 int main(){
7     vehicle v(4);
8     bike b(1);
9     vehicle *vptr = &b;
10
11     cout << v.get_seats() << endl;
12     cout << b.get_seats() << endl;
13     cout << v.get_toll() << endl;
14     cout << b.get_toll() << endl;
15     cout << vptr->get_seats() << endl;
16     cout << vptr->get_toll() << endl;
17     return 0;
18 }
```

6,1 All