CSE 1062 **Fundamentals of Programming**

Lecture #14

Spring 2016

Computer Science & Engineering Program
The School of EE & Computing
Adama Science & Technology University

Outline

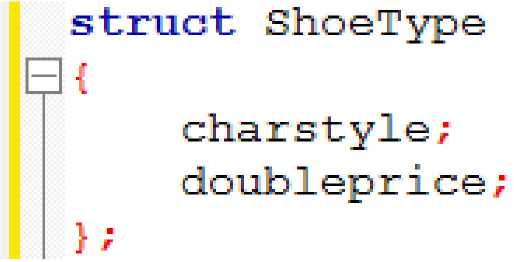
- Structures and Classes Practice
  - Employee Record
  - Practice Exercises 1
  - Continuing the Date Class
  - Elevator Class
  - Practice Exercises 2

Employee Record

- Write a definition for a structure type for records consisting of a person’s wage rate, accrued vacation (which is some whole number of days), and status (which is either hourly or salaried). Represent the status as one of the two char values ’H’ and ’S’. Call the type EmployeeRecord.

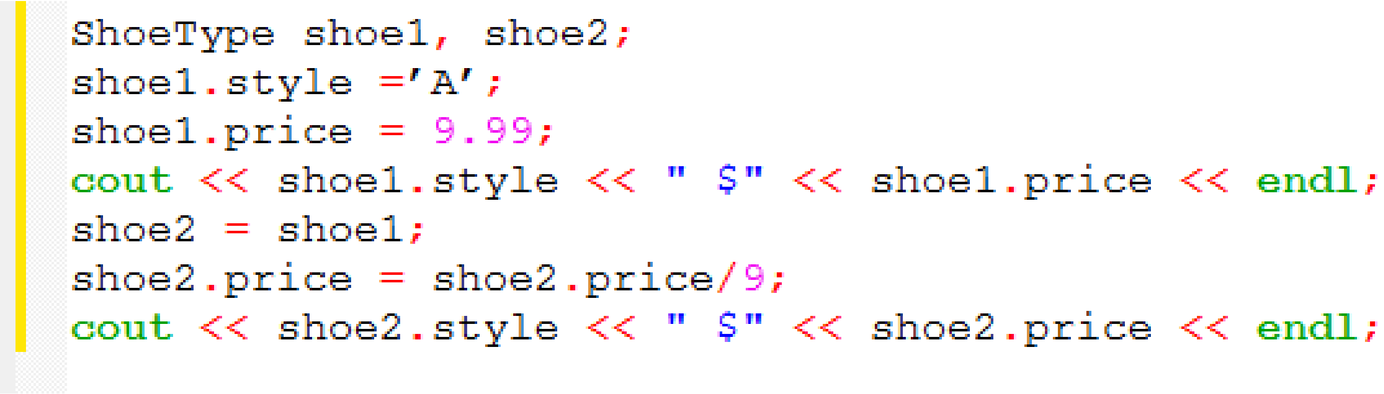
Practice Exercises 1

- Consider the following type definition



```
struct ShoeType
{
    charstyle;
    doubleprice;
};
```

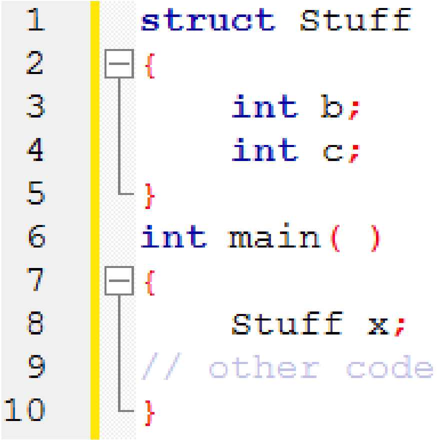
- What will be the output of the following



```
ShoeType shoe1, shoe2;
shoe1.style ='A';
shoe1.price = 9.99;
cout << shoe1.style << " $" << shoe1.price << endl;
shoe2 = shoe1;
shoe2.price = shoe2.price/9;
cout << shoe2.style << " $" << shoe2.price << endl;
```

Practice Exercises 1

- What is the error in the following code



```
struct Stuff
{
    int b;
    int c;
}
int main( )
{
    Stuff x;
// other code
}
```

Continuing the Date Class

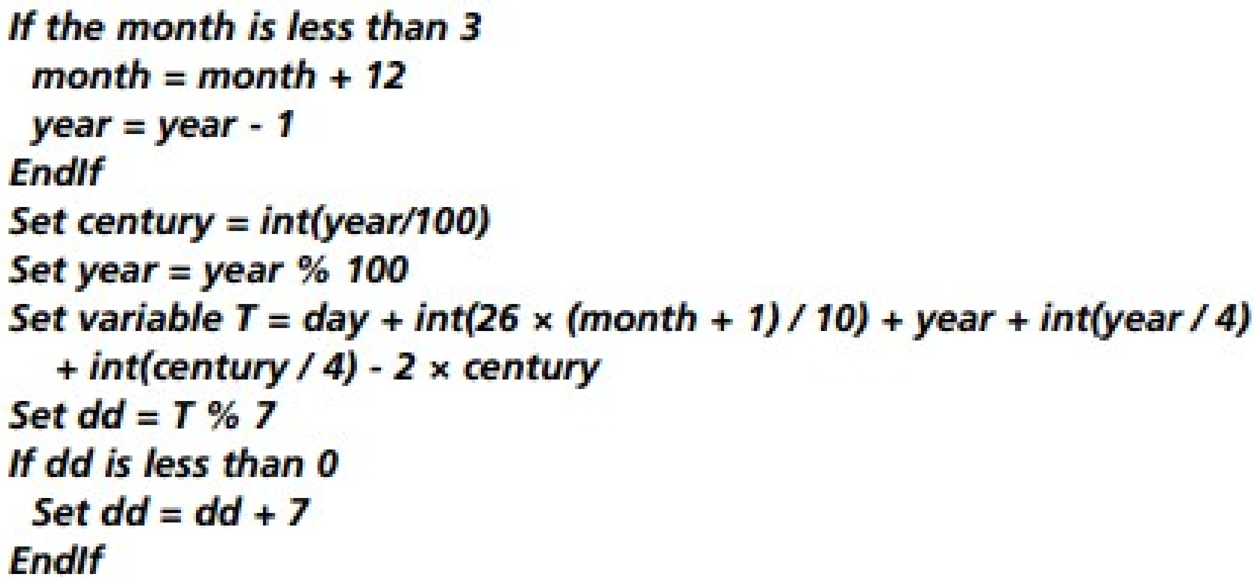
- Add a member function named convert () to the Date class in Lecture 13 that does the following:
- The function should access the month, year, and day data members and return a long integer in the form yyyymmdd that's calculated by using an algorithm

Continuing the Date Class

- yyyymmdd = year * 10000 + month * 100 + day
- For example, if the date is 4/1/2014, the returned value is 20140401.
- (Dates in this form are useful when performing sorts because placing the numbers in numerical order automatically places the corresponding dates in chronological order.)

Continuing the Date Class

- Add a Date class function named dayOfWeek() that returns the day of the week for any date that's provided. Zeller's algorithm is used for determining this information:

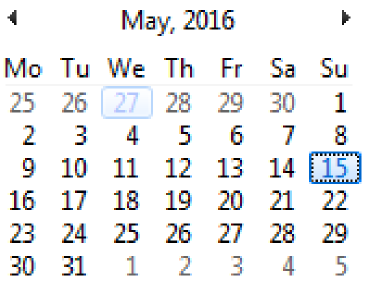


```
If the month is less than 3
  month = month + 12
  year = year - 1
EndIf
Set century = int(year/100)
Set year = year % 100
Set variable T = day + int(26 × (month + 1) / 10) + year + int(year / 4)
    + int(century / 4) - 2 × century
Set dd = T % 7
If dd is less than 0
  Set dd = dd + 7
EndIf
```

Continuing the Date Class

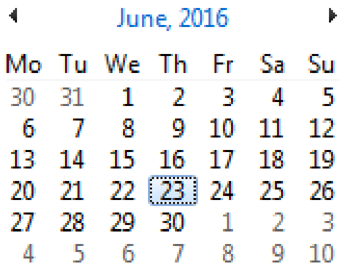
- Using the Zeller's algorithm, the variable dd has a value of 0 if the date is Saturday, 1 if the date is a Sunday, 2 if a Monday, and so on.

- For example, the date 5/15/2016 should return a 1, and the date 6/23/2016 should return a 5



May, 2016

Mo	Tu	We	Th	Fr	Sa	Su
25	26	27	28	29	30	1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31	1	2	3	4	5



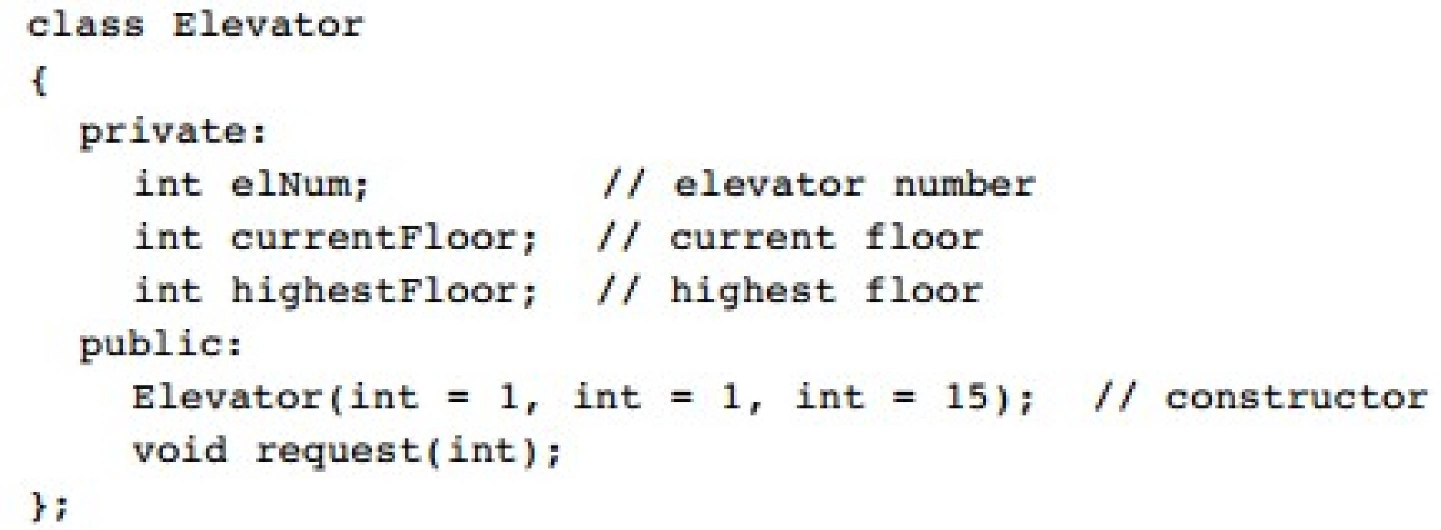
June, 2016

Mo	Tu	We	Th	Fr	Sa	Su
30	31	1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	1	2	3
4	5	6	7	8	9	10

- Put all the additional functions in a complete program and test whether they work correctly

Elevator Class

- Complete the following class by including functions corresponding to the two prototypes listed in the declaration section:



```
class Elevator
{
  private:
    int elNum;           // elevator number
    int currentFloor;   // current floor
    int highestFloor;   // highest floor
  public:
    Elevator(int = 1, int = 1, int = 15);   // constructor
    void request(int);
};
```

Elevator Class

- In this definition,
  - the data member elNum is used to store the elevator’s number,
  - the data member currentFloor is used to stor e the elevator’s current floor position, and the data member highestFloor is used to store th e highest floor the elevator can reach.
- The constructor should allow initialization of an object’s three data members with th e data passed to the constructor when an Elevatorobject is instantiated.

Elevator Class

- The request function should code the following algorithm:

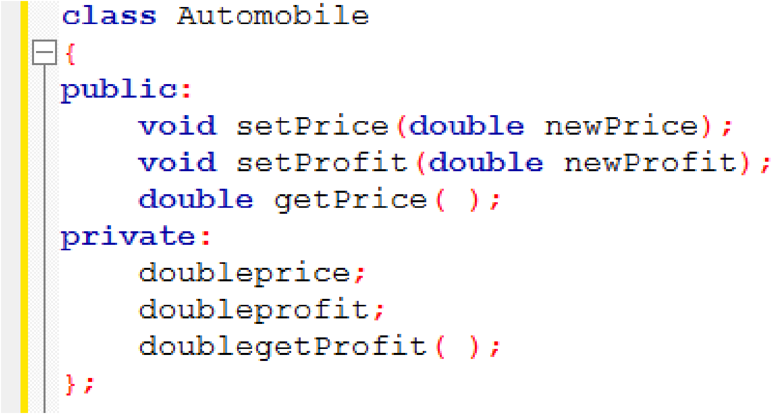
```
If a request is made for a nonexistent floor, a floor higher than the
topmost floor, or the current floor
   Do nothing
ElseIf the request is for a floor above the current floor
   Display the current floor number
   While not at the designated floor
       Increment the floor number
       Display the new floor number
EndWhile
Display the ending floor number
Else // the request must be for a floor below the current floor
  Display the current floor number
  While not at the designated floor
     Decrement the floor number
     Display the new floor number
EndWhile
Display the ending floor number
EndIf
```

Elevator Class

- Include the Elevator class written in a complete program, and verify that all member functions work correctly.

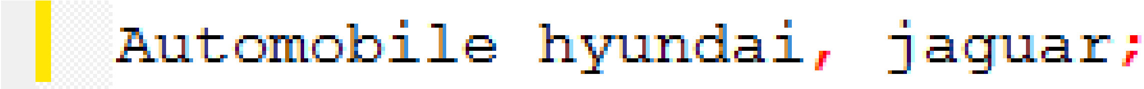
Practice Exercises 2

- Suppose a program contains the following class definition



```
class Automobile
{
public:
    void setPrice(double newPrice);
    void setProfit(double newProfit);
    double getPrice( );
private:
    doubleprice;
    doubleprofit;
    doublegetProfit( );
};
```

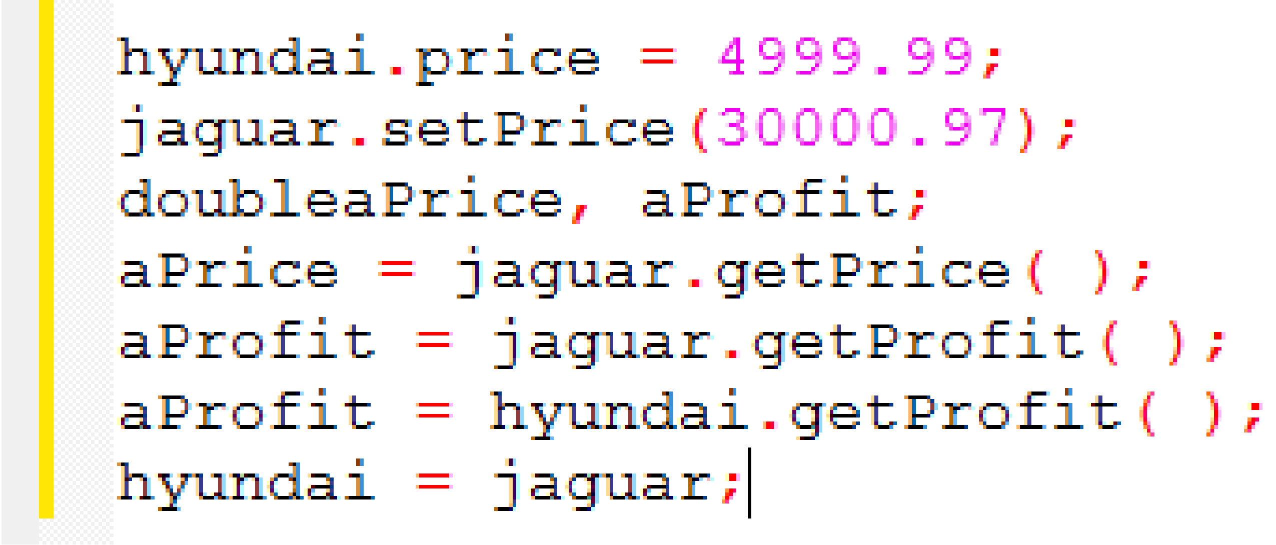
- And suppose the main function contains the following declarations and that the program somehow sets the values of all the member variables to some values



```
Automobile hyundai, jaguar;
```

Practice Exercises 2

- Which of the following statements are then allowed in the main function of your program?



```
hyundai.price = 4999.99;
jaguar.setPrice(30000.97);
doubleaPrice, aProfit;
aPrice = jaguar.getPrice( );
aProfit = jaguar.getProfit( );
aProfit = hyundai.getProfit( );
hyundai = jaguar;
```