**Day 5**

**C++ Constructors and Operators Overloading**

// C++ Default Constructor

/\*

A constructor with no parameters is known as a default constructor.

C++ program to demonstrate the use of default constructor

\*/

#include <iostream>

using namespace ***std***;

// declare a class

*class* Wall{

  private***:***

    double length;

  public***:***

    // create a constructor

*Wall*(){

      // initialize private variables

      length = 5.5;

      cout ***<<*** "Create a wall." ***<<*** endl;

      cout ***<<*** "Length = " ***<<*** length ***<<*** endl;

    }

};

int *main*(){

  // create an object

  Wall wall1;

  return 0;

}

/\*

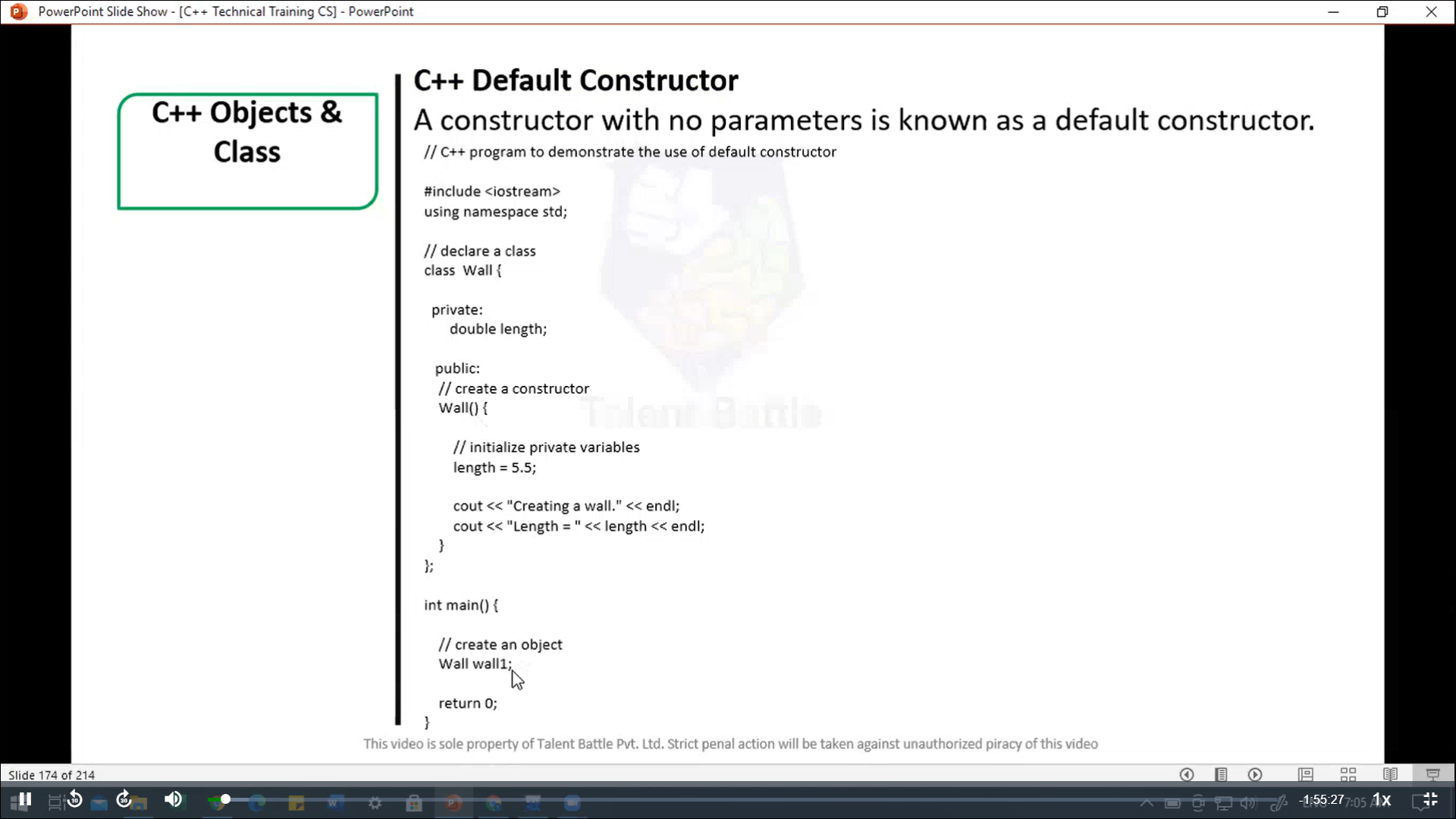
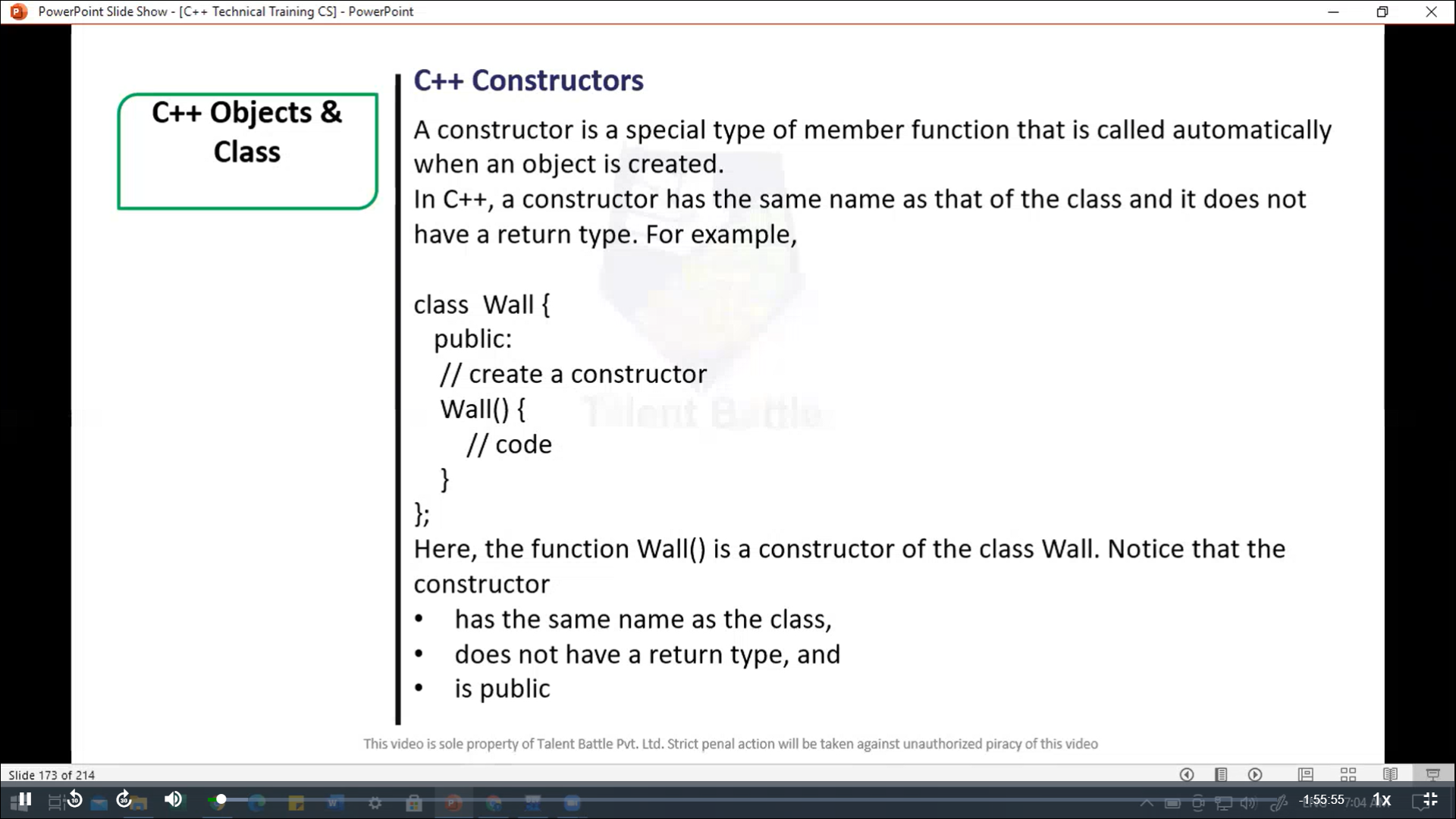
Create a wall.

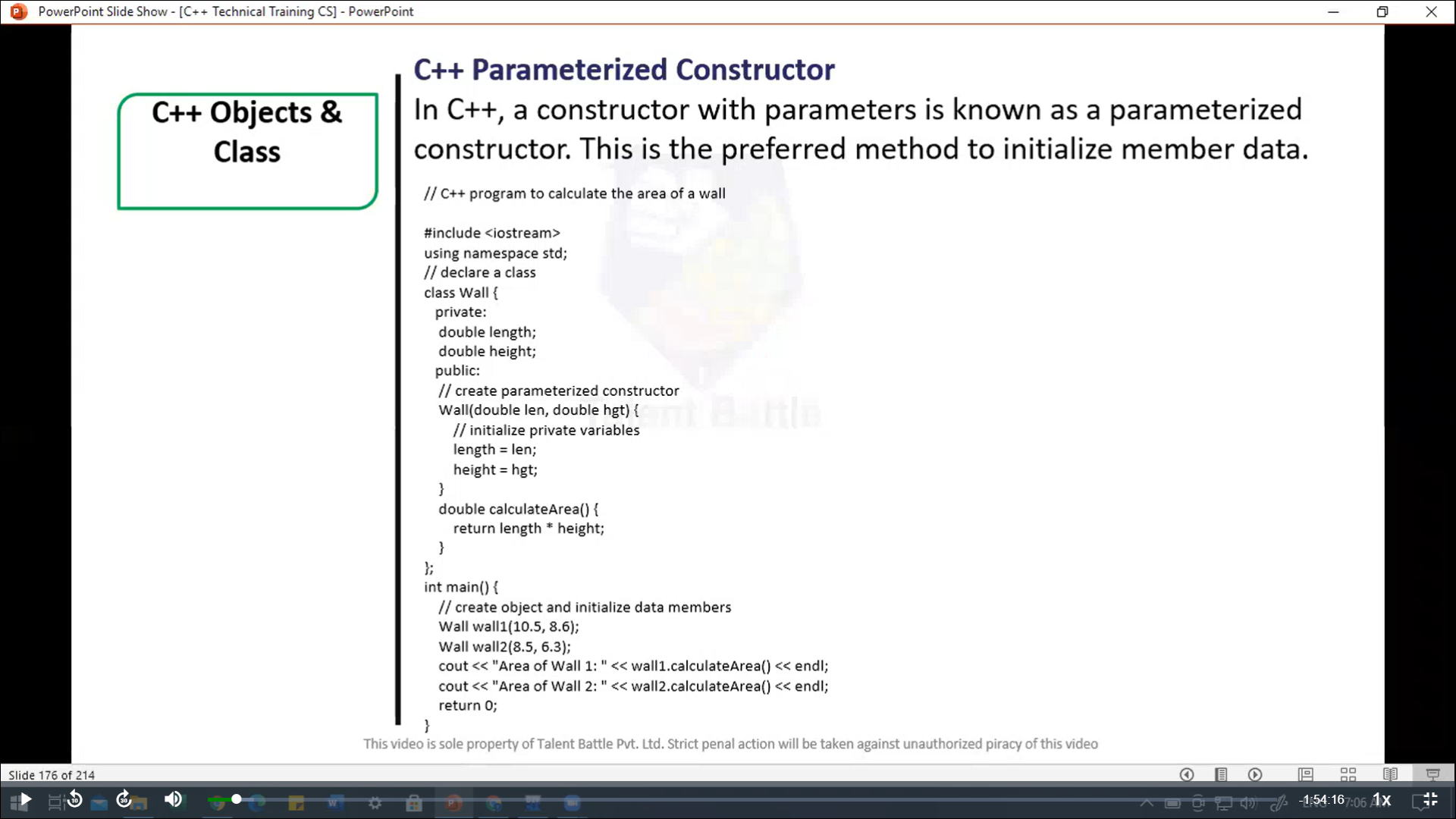
Length = 5.5

--------------------------------

Process exited after 0.1771 seconds with return value 0

\*/

****

****

//==============================================

/\*

C++ Parameterized Constructor

In C++, a constructor with parameters is known as a parameterized

constructor. This is the preferred method to initialize member data.

C++ Program to calculate the area of a wall

\*/

#include <iostream>

using namespace ***std***;

// declare a class

*class* Wall{

  private***:***

    double length;

    double height;

  public***:***

    // creae parameterized constructor

*Wall*(double ***len,*** double ***hgt***){

      // initialize private variables

      length = ***len***;

      height = ***hgt***;

    }

    double *calculateArea*(){

      return length \* height;

    }

};

int *main*(){

  // create object and initialize data members

  Wall wall1(10.5***,*** 8.6);

  Wall wall2(8.5***,*** 6.3);

  cout *<<* "Area of Wall1: " *<<* wall1***.****calculateArea*() *<<* *endl*;

  cout *<<* "Area of Wall2: " *<<* wall2***.****calculateArea*() *<<* *endl*;

  return 0;

}

/\*

Area of Wall1: 90.3

Area of Wall2: 53.55

--------------------------------

Process exited after 0.1177 seconds with return value 0

Press any key to continue . . .

\*/

//==================================================

// C++ Copy Constructor

// The copy constructor in C++ is used to copy data of one object to another

#include <iostream>

using namespace ***std***;

// declare a class

*class* Wall{

  private***:***

    double length;

    double height;

  public***:***

    // parameterized constructor

*Wall*(double ***len,*** double ***hgt***){

      // initialize private variables

      length = ***len***;

      height = ***hgt***;

    }

    // copy constructor with a wall object as parameter

*Wall*(Wall &***obj***){

      // initialize private variables

      length = ***obj.***length;

      height = ***obj.***height;

    }

    double *calculateArea*(){

      return length \* height;

    }

};

int *main*(){

  // Create an object of Wall class

  Wall wall1(10.5***,*** 8.6);

  // print area of wall1

  cout *<<* "Area of Wall 1: " *<<* wall1***.****calculateArea*() *<<* *endl*;

  // Copy contents of wall1 to another object wall2

  Wall wall2 = wall1;

  // print area of wall2

  cout *<<* "Area of Wall 2: " *<<* wall2***.****calculateArea*() *<<* *endl*;

  return 0;

}

/\*

Area of Wall 1: 90.3

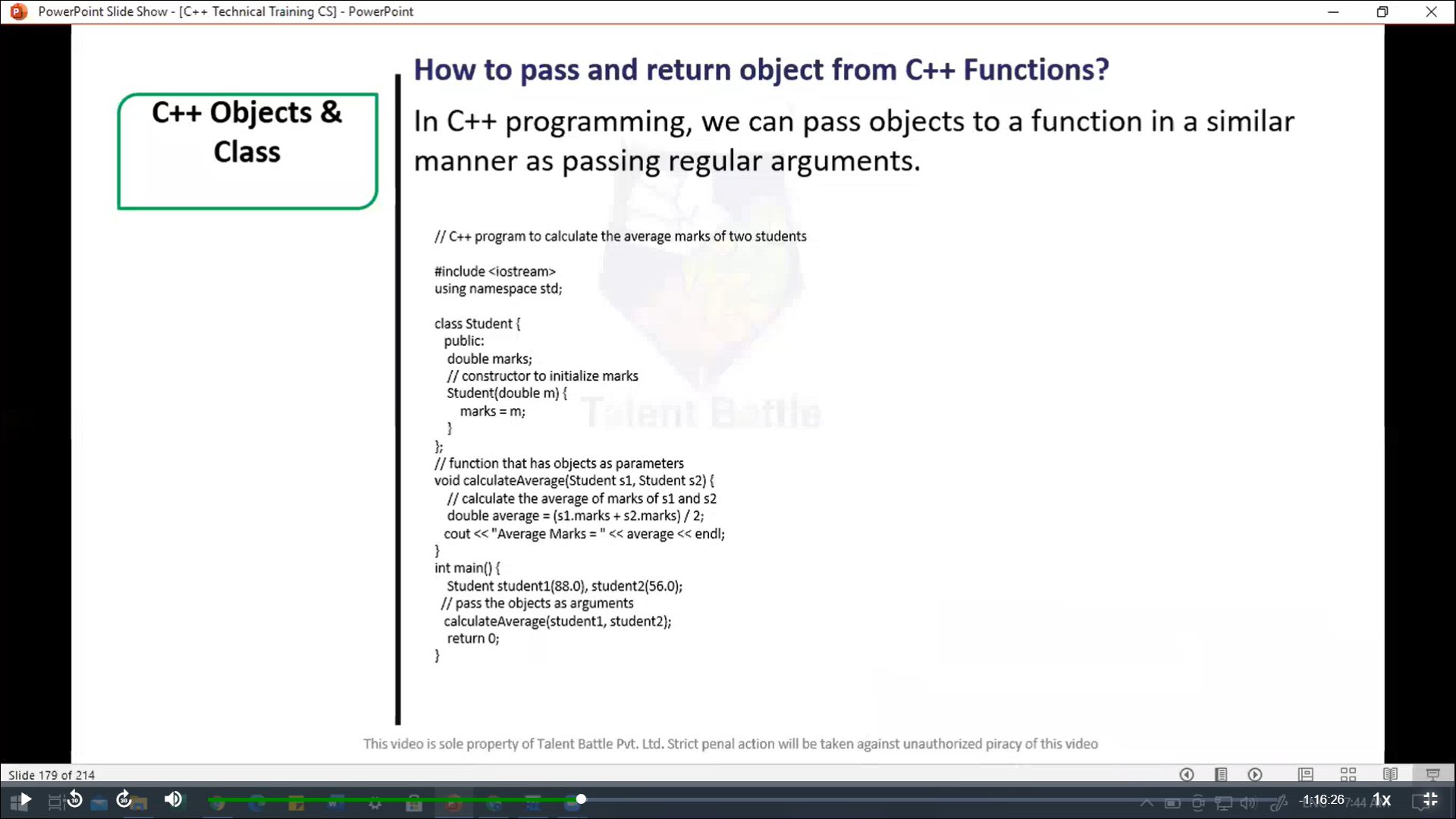
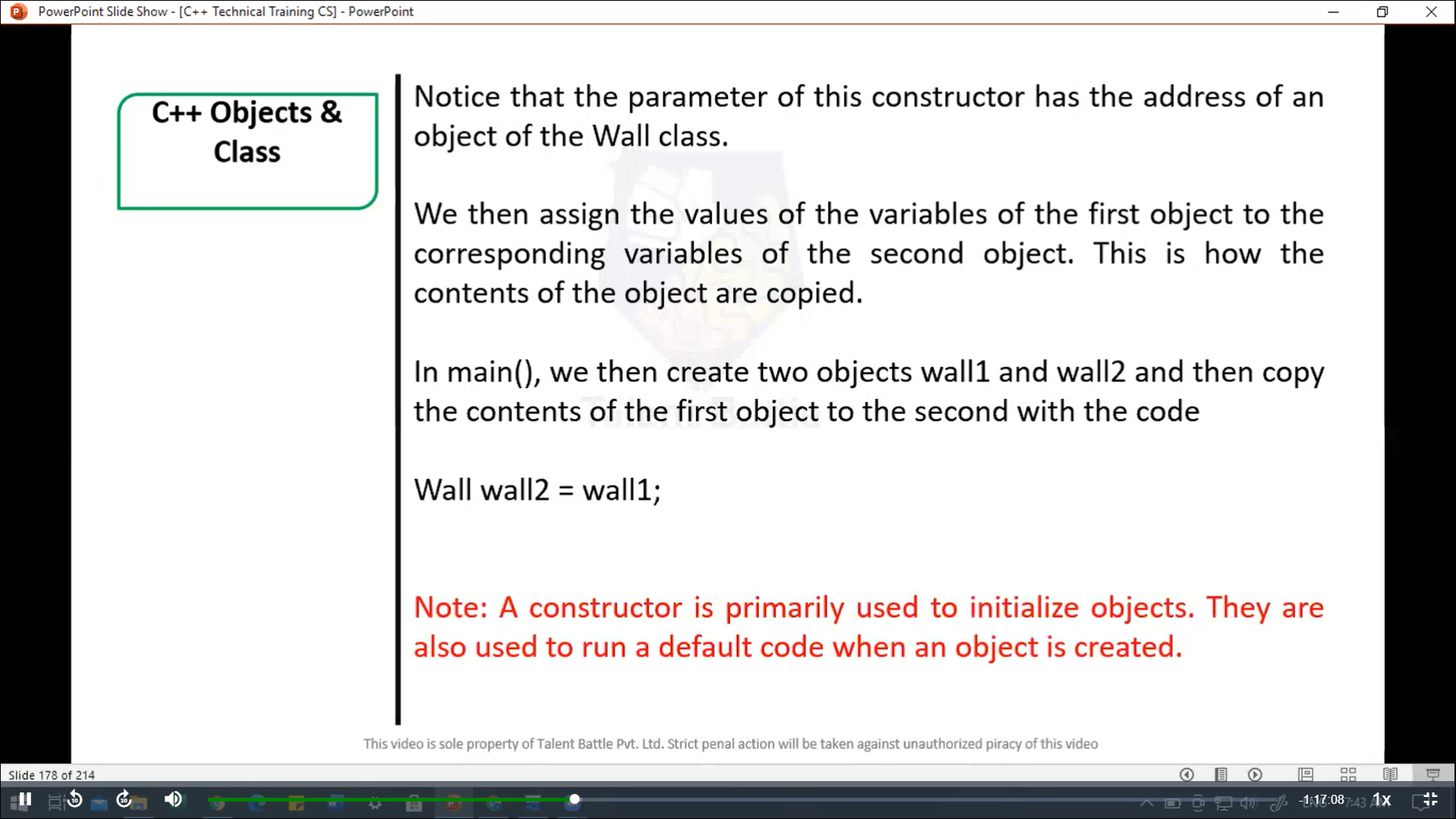
Area of Wall 2: 90.3

--------------------------------

Process exited after 0.1467 seconds with return value 0

Press any key to continue . . .

\*/

****

//=======================================================

/\*

How to pass and return object from C++ functions?

In C++ programming, we can pass objects to a function in a

similar manner as passing  regular arguments.

\*/

// C++ program to calculate the average marks of two students

#include<iostream>

using  namespace ***std***;

*class* Student{

  public***:***

    double marks;

    // constructor to initialize marks

*Student*(double ***m***){

      marks = ***m***;

    }

};

// function that has object as parameters

void *calculateAverage*(Student ***s1,*** Student ***s2***){

  // calculate the average of marks of s1 and s2

  double average = (***s1.***marks + ***s2.***marks) / 2;

  cout *<<* "Average Marks = " *<<* average *<<* *endl*;

}

int *main*(){

  Student student1(88.0)***,*** student2(56.0);

  // pass the objects as arguments

*calculateAverage*(student1***,*** student2);

  return 0;

}

/\*

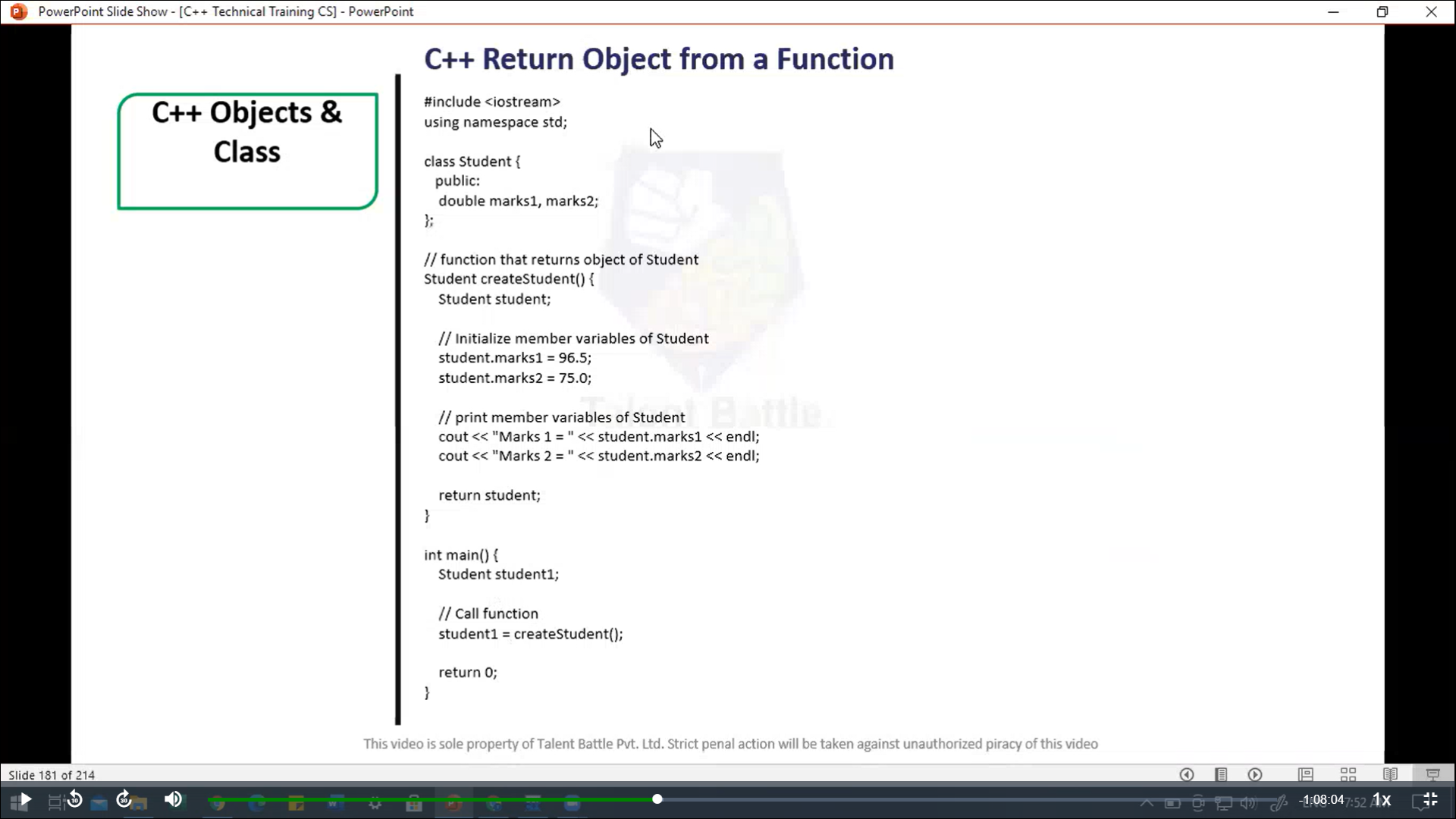
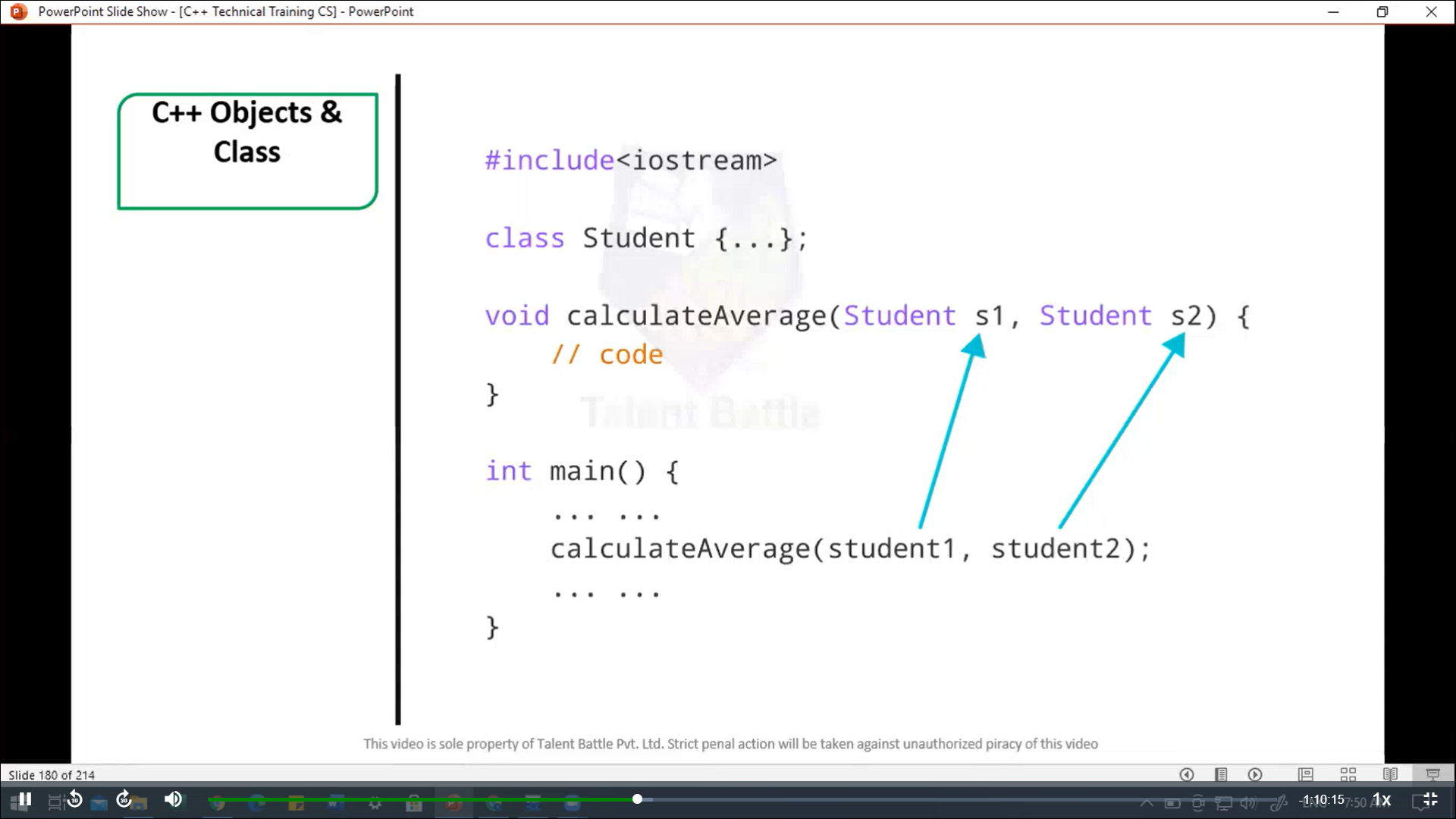
Average Marks = 72

--------------------------------

Process exited after 0.1455 seconds with return value 0

Press any key to continue . . .

\*/

****

//============================================

// C++ Return Object from a Function

#include<iostream>

using namespace ***std***;

*class* Student{

  public***:***

    double marks1***,*** marks2;

};

// function that returns object of Student

Student *createStudent*(){

  Student student;

  // initialize member variables of Student

  student***.***marks1 = 96.5;

  student***.***marks2 = 75.0;

  // print memeber variables of Student

  cout *<<* "Marks 1 = " *<<* student***.***marks1 *<<* *endl*;

  cout *<<* "Marks 2 = " *<<* student***.***marks2 *<<* *endl*;

  return student;

}

int *main*(){

  Student student1;

  // call function

  student1 *=* *createStudent*();

  return 0;

}

/\*

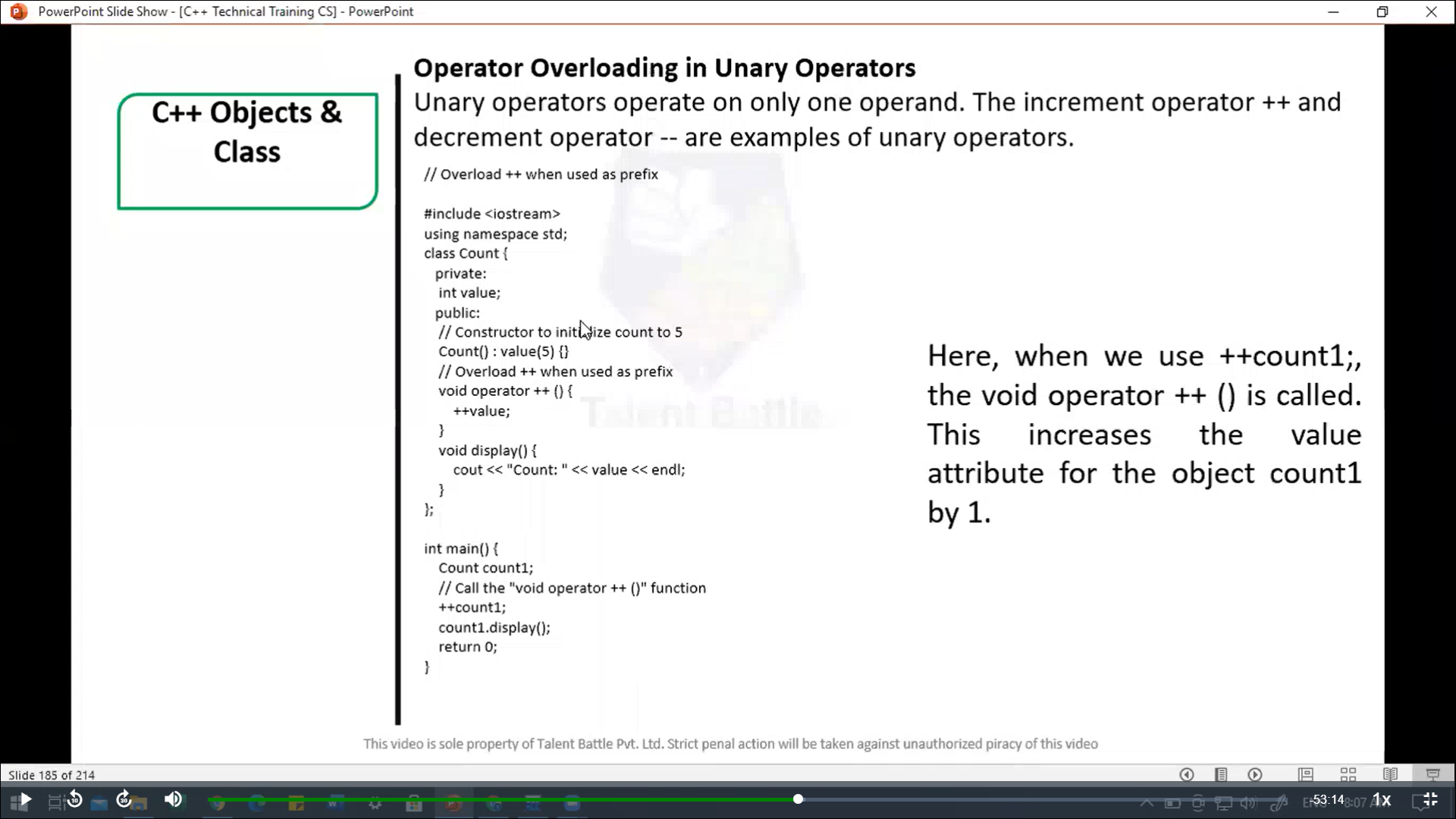
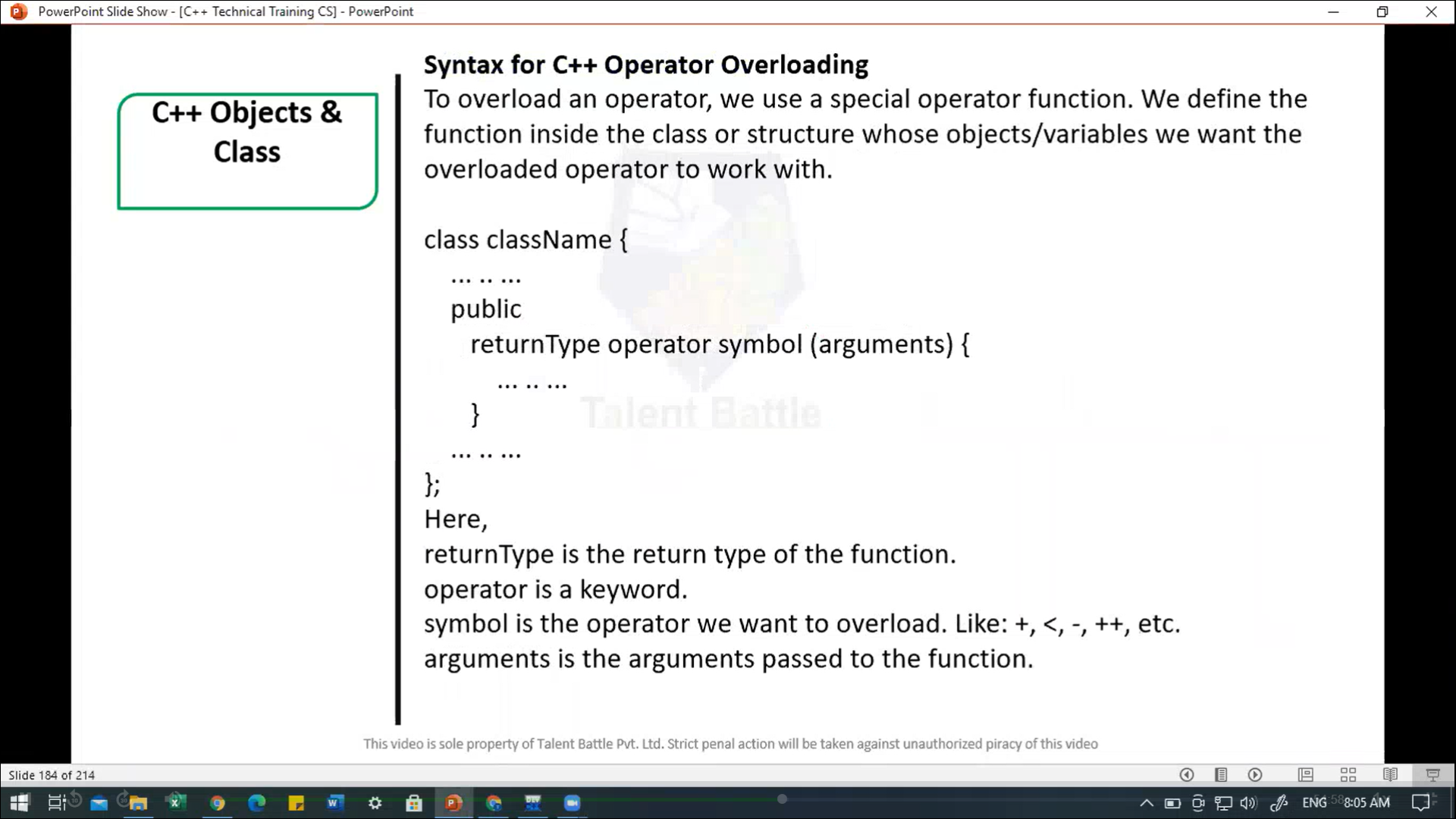
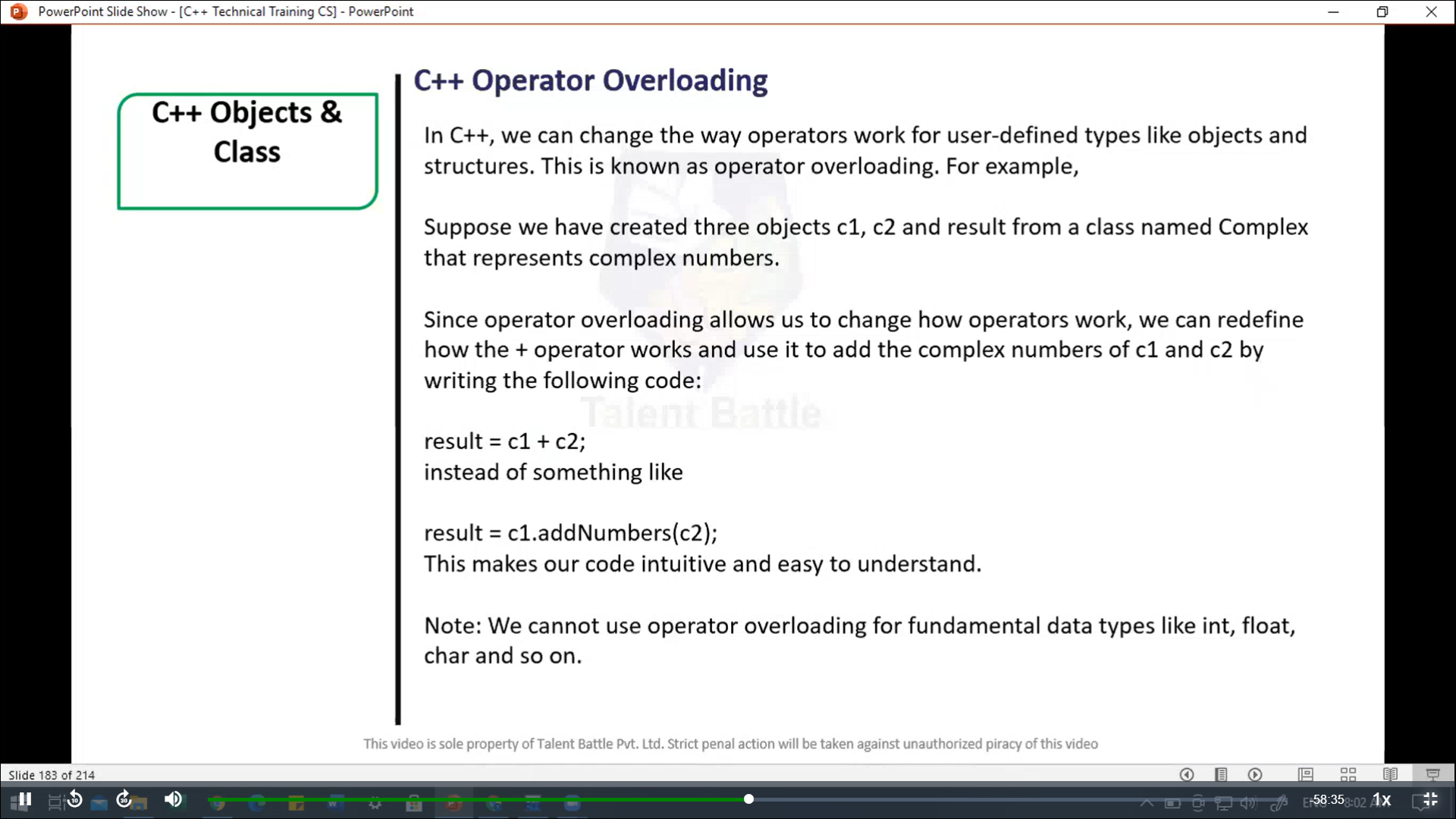
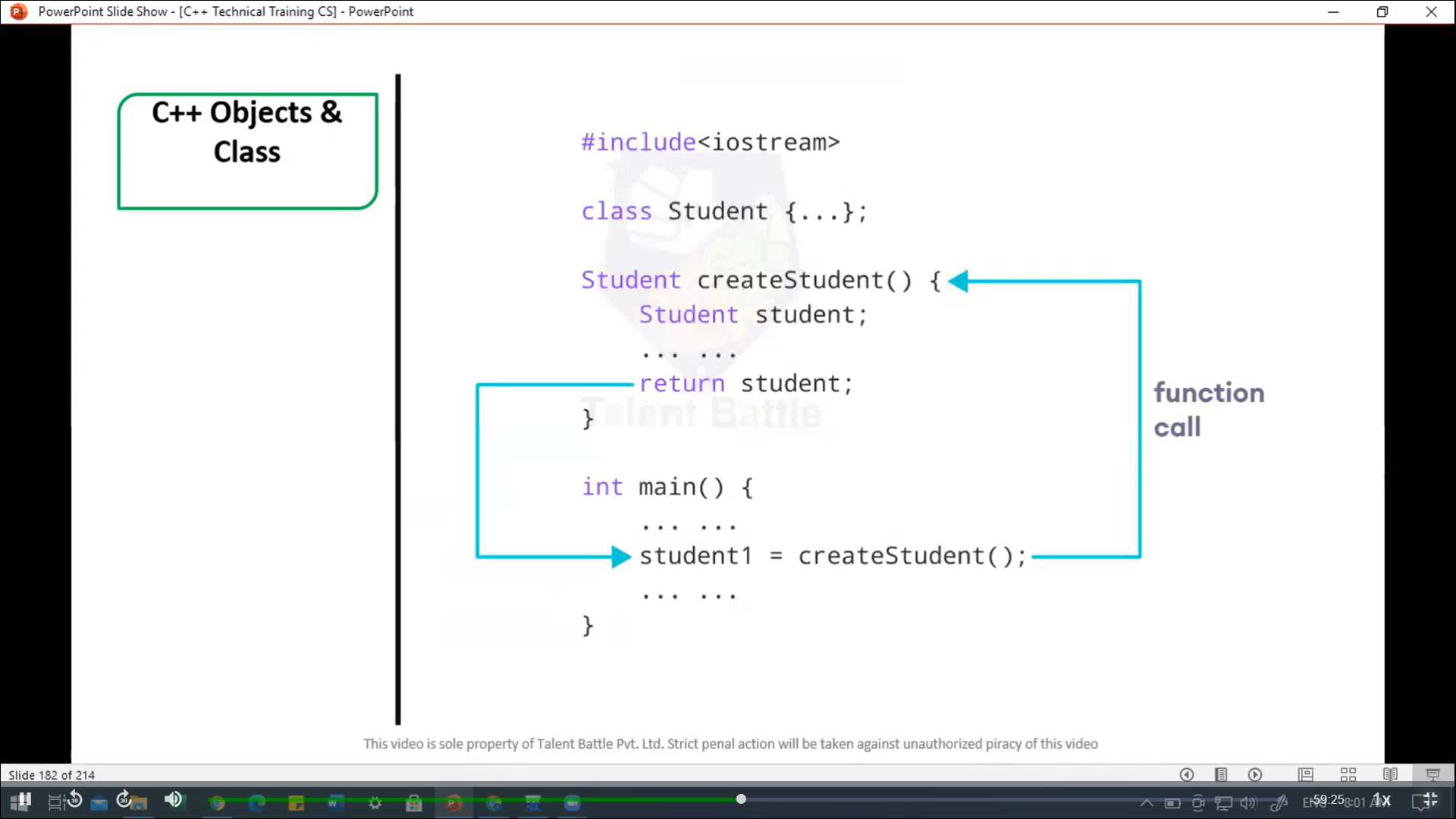
Marks 1 = 96.5

Marks 2 = 75

--------------------------------

Process exited after 0.06273 seconds with return value 0

\*/

****

//====================================================

/\*

Operator Overloading in Unary Operators

--> Unaryoperatrs operate on only one operand. The increment

operator ++ and decrement operator -- are example of unary

operators.

Here, when we use ++count1;, the void operator ++ () is

called.

This increasees the value attribute for the object

coutn1 by 1.

\*/

// Overload ++ when used as prefix

#include <iostream>

using namespace ***std***;

*class* Count {

  private***:***

    int value;

  public***:***

    // Constructor to initialize count to 5

*Count*()***:*** value(5){}

    // Overload ++ when used as prefix

    void *operator ++* (){

      ++value;

    }

    void *display*(){

      cout *<<* "Count: " *<<* value *<<* *endl*;

    }

};

int *main*(){

  Count count1;

  // Call the "void operator ++ ()" function

*++*count1;

  count1***.****display*();

  return 0;

}

/\*

Count: 6

--------------------------------

Process exited after 0.06413 seconds with return value 0

\*/