1) explanation about each code line

#include <iostream>

/*

An important standard library is iostreams, which allow you to read from files and the keyboard, and to write to files and the display.

The preprocessor will find the iostream header file (often in a subdirectory called “include”) and insert it.

Do you know when stdio.h is callled in a C Program ad who calls

it? */

using namespace std;//All of the Standard C++ libraries are wrapped in a single namespace, which is std (for “standard”).

int main() {

// In C++, main( )always has return type of int.

int i;

cout << "This is output.\n"; // To be read as "Send the string “This is output.” to the object called cout".

// output opreator

/*

The operator cout is an object

Belongs to output stream class

Used to perform write operations on the output devices e.g. screen,

disk etc.

Used with left shift operator (<<), called insertion or put-to operator or output

operator

Syntax

cout « "variable";

Variable may be built in data type/string/constant/blank character

It also supports cascaded output operations

Does not require explicit Format specification (unlike printf that needs “string”)

*/

// this is a single line comment /* you can still use C style comments */

// input a number using >>

cout << "Enter a number: "; //input operator

cin >> i;

/*

The operator cin is an object

Belongs to input stream class

Used to perform read operations from the input devices e.g. keyboard

Used with right shift operator (>>), called extraction / get /input operator Syntax

cin » ""variable" ";

Variable may be built in data type/string/blank character

It should have at least one argument

It also supports cascaded input operations

Does not require explicit Format specification (unlike scanff that needs “string”)

*/
// now, output a number using <<

cout <<" The number given is : " << i << "\n";

return 0;//int return from main()

}

==========

= ==========

2 loops and return function

==========

= ==========

#include <iostream>

/* An important standard library is iostream, which allows you to read from files

and the keyboard, and to write to files and the display.

The preprocessor will find the iostream header file and insert it. */

using namespace std;

// All of the standard C++ libraries are wrapped in a single namespace, which is std (short for "standard").

int main()

// In C++, the main() function always has a return type of int, which indicates the exit status of the program.

{

int a = 6;

// Declares an integer variable `a` and initializes it to 6.

int b = 0;

// Declares another integer variable `b` and initializes it to 0.

```
    while (a < 10)
    // Starts a while loop that will continue running as long as the condition `a < 10` is true.
    {
        a = a / 12 + 1;
        // Updates `a` by dividing it by 12, then adding 1. This operation causes an infinite loop as `a` doesn't grow beyond 10.


        a += b;
        // Adds the value of `b` to `a` (same as `a = a + b`). Here, `b` is 0, so `a` remains unchanged in value.


        // cout << a << b;
        // Uncommenting this would print the current values of `a` and `b` to the display.
        // it is an object belongs to ostream class "<<"
        // refers to bits that are sending from memory to desktop.
    }


    cout << a;
    // Outputs the value of `a` after the loop ends (but the loop is infinite, so this line will never execute).
    // it is an object belongs to ostream class "<<"
    // refers to bits that are sending from memory to desktop.


    return 0;
    // Returns 0 to indicate that the program executed successfully.
}
```

/* io is input and output stream

std standard */

/*

it is an object belongs to ostream class"<<" refres to bits are sending from

memory to desktop

it is an object belongs to istream">>"refers to bits from key board to memory

*/

==========

= ==========

3 CREATING THE STUDENT PROGRAM

==========

= ==========

/*

EXERCISE TO CREATE A PROGRAM BY CREATING

1)GROUPS ITEMS PROPERTIES as:VARIABLES

2)BEHAVIOR as: FUNCTIONS

GROUP STUDENTS()

NAME:

SEMISTER:

NO_OF_STUDENTS:

TOTAL_MARKS_SECURED:

FINALLAY ()

:-AVG_MARKS_SECURED()

*/

#include<iostream> // preprocessor directive

/* an important standard library is iostream, which allow you to read from files

and the keyboard, and to write to fies and the display.*/

using namespace std; //standard library

int main(){

// main alwys has return type of int in C++

// Entry point of the program, always returns an int.

string name; //member variable decleration

int clas;// Variable to store the student's class.

int i,marks; // Variables to store loop index (if used) and marks.

cout << "enter your name: "; //it is an object belongs to ostream (output stream)class"<<" refers to bits are sending from memory to desktop

cin>> name;// it is an object belongs to istream class">>" refers to bits coming from

cout << "enter your class";//it is an object belongs to ostream and "<<" refers to bits from memory to desk

cin >> class; // it is an object belongs to istream class">>" refers to bits coming from

cout << "enter marks";//it is an object belongs to ostream and "<<" refers to bits from memory to desk

cin >> marks;// it is an object belongs to istream class">>" refers to bits coming from

//cout<< "Do you want to see the result (y/n)";

// cin>> i;

```
//        if( i== "y"|| i=="Y"):

  cout<< "name:" <<name<<" class "<<clas<< " marks "<<marks;

   // else:

    //     cout<<"thank you";

}

/*

class students{

  string name;

  int sem;

  int stud_no;

  int tot_marks;

    public: void set(){

          cout <<"Enter your name:";

          cin  >>
```

```
 }

};

*/
```

==========
= ==========

4 CREATING CLASS WITH VOID FUNCTIONS AND WITH SET&DISPLAY

==========
= ==========

```
#include<iostream>   //compulsery for all the codes
/*EXERCISE  TO  CREATE{
CLASS NAME: TEST
member variable: int a
member function : void set(),void Display()
 */

using namespace std;
class test{                    // creation of class
int a;                           //member variable declared
public:                 // global variable
 void set(){                    //member functions
   cout<<"enter the value of A:";
    cin >>a;
 }
 void display(){                //member function  {void funtion will not
```

```
occupy memory}
        cout<<"the value of A is:"<<a<<endl;  //endl is end line


 }
};       // body of the class(test) is closed

int main(){   //outputs can be taken in main function only
 test t1; //creation of object/variable ... Memory is allocated here
        // for the class test
 t1.set();   //transfering the {set funtion output in fl object and displaying
it}
 t1.display(); //display funtion output in tl;
 return 0;
}
```

==========

= ==========

5 STRING INPUT

==========

= ==========

```
#include <iostream>

// Includes the iostream library, which allows input and
output operations.


using namespace std;

// Allows using names from the standard namespace
without prefixing them with "std::".
```

int main() {

// Entry point of the program where execution begins.

string mystring = "Hello, world!";

// Declares a string variable named 'mystring' and initializes it with "Hello, world!".

cout << mystring;

// Outputs the value of 'mystring' to the desktop (it is an object belonging to the ostream class; "<<" refers to bits being sent from memory to the desktop).

return 0;

// Indicates successful program termination to the operating system.

}

==========
= ==========

6 ANIMAL DATAENTRY

==========
= ==========

#include <iostream> // Preprocessor directive to include the iostream library for input and output operations.

using namespace std; // Allows using standard library functions and objects without the `std::` prefix.

int main() // The main() function, the entry point of the program, has a return type of int.

{

string ani1, ani2; // Declares two string variables to store the names of two animals.

string col1, col2; // Declares two string variables to store the colors of the animals.

int age1, age2; // Declares two integer variables to store the ages of the animals.

int w1, w2, h1, h2; // Declares integer variables to store the weights and heights of the animals.

cout << "enter animal1 name:"; // (it is an object belongs to ostream class "<<" refers to bits sent from memory to desktop)

cin >> ani1; // (it is an object belongs to istream class ">>" refers to bits from keyboard to memory)

cout << "enter animal1 color1 :"; // Prompts the user to enter the color of the first animal.

cin >> col1; // Reads the color of the first animal from the user input.

cout << "enter animal1 age:"; // Prompts the user to enter the age of the first animal.

cin >> age1; // Reads the age of the first animal from the user input.

cout << "enter animal2 name:"; // Prompts the user to enter the name of the second animal.

cin >> ani2; // Reads the name of the second animal from the user input.

cout << "enter animal2 color1 :"; // Prompts the user to enter the color of the second animal.

cin >> col2; // Reads the color of the second animal from the user input.

cout << "enter animal2 age:"; // Prompts the user to enter the age of the second animal.

cin >> age2; // Reads the age of the second animal from the user input.

// Outputs details about the first animal including its name, color, and age.

cout << "the 1st animal is " << ani1 << " color is { " << col1 << " } age { " << age1 << " }\n";

// Outputs details about the second animal including its name, color, and age.

cout << "the 2nd animal is " << ani2 << " color is { " << col2 << " } age { " << age1 << " }\n";

cout << "weight of animal " << ani1 << " is "; // Prompts the user to enter the weight of the first animal.

cin >> w1; // Reads the weight of the first animal from the user input.

cout << "weight of animal " << ani2 << " is "; // Prompts the user to enter the weight of the second animal.

cin >> w2; // Reads the weight of the second animal from the user input.

return 0; // Indicates successful program termination.

}

==========

==========

7

==========

==========

#include <iostream>

// Includes the iostream library for input and output operations

using namespace std; // Allows usage of standard library features without prefixing them with std::

class student { // Defines a class named 'student' to represent a student entity

string name, clas, place; // Private member variables to store name, class, and place of the student

int roll, age; // Private member variables to store roll number and age of the student

public: // Public access specifier to allow access to member functions outside the class

void set() { // Member function to input values for the student cout << **"enter your name: ";** //it is an object belongs to ostream (output stream)class"<<" refers to bits are sending from memory to desktop

cin >> name; // it is an object belongs to istream class">>" refers to bits coming from

cout << "enter roll: "; //it is an object belongs to ostream (output stream)class"<<" refers to bits are sending from memory to desktop
cin >> roll; // it is an object belongs to istream class">>" refers to bits coming from

cout << "enter class: "; //it is an object belongs to ostream (output stream)class"<<" refers to bits are sending from memory to desktop
cin >> clas; // it is an object belongs to istream class">>" refers to bits coming from

cout << "enter age: "; //it is an object belongs to ostream (output stream)class"<<" refers to bits are sending from memory to desktop
cin >> age; // it is an object belongs to istream class">>" refers to bits coming from

cout << "enter place: "; //it is an object belongs to ostream (output stream)class"<<" refers to bits are sending from memory to desktop
cin >> place; // it is an object belongs to istream class">>" refers to bits coming from

void display() { // Member function to display the values of the student // Output the student details in a formatted way
cout << "name: " << name << " roll: " << roll
<< " class: " << clas << " age: " << age
<< " place: " << place << "\n";

//it is an object belongs to ostream (output stream)class"<<" refers to bits are sending from memory to desktop

}

};

int main() { // Main function where program execution begins student s1; // Create an object 's1' of class 'student'

s1.set(); // Call the set() function to input student details

s1.display(); // Call the display() function to output student details

return 0; // Return 0 to indicate successful execution of the program }

==========

==========

8 student database with set display fuction along with

==========

==========

#include <iostream>
// Header file for input and output operations

using namespace std;

// Define a class named 'student' to manage student information and operations

class student {

string name, sem; // Variables to store the student's name and

semester int no_sub, tot_marks; // Variables to store the number of subjects and total marks

```
public:
// Function to set the student details
void set() {
cout << "enter your name: ";
// it is an object belongs to ostream (output stream) class "<<" refers to
bits are sending from memory to desktop
cin >> name;
// it is an object belongs to istream class ">>" refers to bits coming from

cout << "enter semester: ";
// it is an object belongs to ostream (output stream) class "<<" refers to
bits are sending from memory to desktop
cin >> sem;
// it is an object belongs to istream class ">>" refers to bits coming from

cout << "enter no_of_subjects: ";
// it is an object belongs to ostream (output stream) class "<<" refers to
bits are sending from memory to desktop
cin >> no_sub;
// it is an object belongs to istream class ">>" refers to bits coming from

cout << "enter total_marks: ";
// it is an object belongs to ostream (output stream) class "<<" refers to
bits are sending from memory to desktop
cin >> tot_marks;
// it is an object belongs to istream class ">>" refers to bits coming from
}
```

```
// Function to calculate and return the average marks
float average() {
int avg; // Variable to store the average marks
avg = tot_marks / no_sub; // Calculate the average by dividing
total marks by the number of subjects
return avg; // Return the calculated average
}

// Function to display the calculated average marks
void display() {
cout << "average :" << average() << endl;
// it is an object belongs to ostream (output stream) class "<<" refers to
bits are sending from memory to desktop
}
};

int main() {
student s1, s2; // Create two objects of the 'student' class

// Set details for the first student
s1.set();

// Set details for the second student
s2.set();

// Display the average marks for the first student
s1.display();
```

// Display the average marks for the second student

s2.display();

return 0; // Return 0 to indicate successful program execution

}

==========

==========

9 complex numbers (a+bi) addition &multiplication

==========

==========

#include <iostream>
// Header file for input and output operations

using namespace std; // Allows the use of standard namespace to avoid writing std:: repeatedly

// Define a class to represent complex numbers

class complex {

public:

float real, imag; // Member variables to store the real and imaginary parts of the complex number

// Method to set the real part of the complex number

void setreal(float r) {

// Passing objects (float r) as parameters real = r;

```
}

// Method to set the imaginary part of the complex number
void set_imaginary(float im) {
imag = im;
}

/*
Function name: add
Parameters: `next` of type complex
Returns: A complex number as the result of addition
(ans) */
complex add(complex next) { // Passing another object as a
parameter complex ans; // Temporary object to store the result
ans.real = real + next.real; // Adding real parts
ans.imag = imag + next.imag; // Adding imaginary
parts return ans; // Returning the result
}

// Function to calculate the product of two complex numbers
complex prod(complex next) {
complex ans; // Temporary object to store the result
ans.real = real * next.real - imag * next.imag; // Formula for real
part of product
ans.imag = real * next.imag + imag * next.real; // Formula for
imaginary part of product
return ans; // Returning the result
}
```

```
// Function to display the complex number in a+bi or a-bi format
void displayresult() {
if (imag >= 0) {
cout << real << "+" << imag << "i" << endl;
// it is an object belongs to ostream (output stream) class "<<" refers to
bits are sending from memory to desktop
} else {
cout << real << "-" << imag << "i" << endl;
// it is an object belongs to ostream (output stream) class "<<" refers to
bits are sending from memory to desktop
}
}
};
int main() {
complex num1, num2, result; // Creating two complex number
objects and one result object

float t_real, t_imag; // Temporary variables to hold input for real
and imaginary parts

// 1st complex number
cout << "Enter real number for the 1st complex number: ";
// it is an object belongs to ostream (output stream) class "<<" refers
to bits are sending from memory to desktop
cin >> t_real;
// it is an object belongs to istream class ">>" refers to bits coming from
num1.setreal(t_real); // Set the real part of the first complex number
```

cout << "Enter imaginary number for the 1st complex number: ";

// it is an object belongs to ostream (output stream) class "<<" refers to bits are sending from memory to desktop

cin >> t_imag;

// it is an object belongs to istream class ">>" refers to bits coming from num1.set_imaginary(t_imag); // Set the imaginary part of the first complex number

// 2nd complex number

cout << "\nEnter real number for the 2nd complex number: ";

// it is an object belongs to ostream (output stream) class "<<" refers to bits are sending from memory to desktop

cin >> t_real;

// it is an object belongs to istream class ">>" refers to bits coming from num2.setreal(t_real); // Set the real part of the second complex number

cout << "Enter imaginary number for the 2nd complex number: ";

// it is an object belongs to ostream (output stream) class "<<" refers to bits are sending from memory to desktop

cin >> t_imag;

// it is an object belongs to istream class ">>" refers to bits coming from

num2.set_imaginary(t_imag);

// Set the imaginary part of the second complex number

// Addition of two complex numbers

result = num1.add(num2);

// Call the add function

cout << "\nSum of two complex numbers is: ";

// it is an object belongs to ostream (output stream) class "<<" refers to bits are sending from memory to desktop

result.displayresult(); // Display the result of addition

// Multiplication of two complex numbers

result = num1.prod(num2); // Call the prod function

cout << "\nProduct of two complex numbers is: ";

// it is an object belongs to ostream (output stream) class "<<" refers to bits are sending from memory to desktop

result.displayresult(); // Display the result of multiplication

return 0; // Indicate successful execution

}

/*

A complex number is represented as a+bi in float format, where `a` is the real part, and `b` is the imaginary part.

For example, 5i+4.7j = complex number.

*/

==========

==========

10 food_data representation

==========

==========

```
#include <iostream>
// Includes the standard input/output stream library

/*Group-Food
Pizza, Chocolate, Ice Cream
Color, Taste, Cost
Display( )*/

using namespace std; // Allows direct use of standard
namespace components

// Class representing food items
class food {
string name, color, taste; // Attributes to store name, color, and taste
of the food
int cost; // Attribute to store the cost of the food
public:
// Function to input food details from the user
void set() {
cout << "Enter name of the food you want to display: ";

// it is an object belongs to ostream (output stream) class "<<" refers
to bits are sending from memory to desktop
cin >> name;
// it is an object belongs to istream class ">>" refers to bits coming from

cout << "Enter color of it: ";
// it is an object belongs to ostream (output stream) class "<<" refers to
bits are sending from memory to desktop
cin >> color;
```

```
// it is an object belongs to istream class ">>" refers to bits coming from

cout << "Enter taste: ";
// it is an object belongs to ostream (output stream) class "<<" refers to bits are sending from memory to desktop
cin >> taste;
// it is an object belongs to istream class ">>" refers to bits coming from

cout << "Enter cost: ";
// it is an object belongs to ostream (output stream) class "<<" refers to bits are sending from memory to desktop
cin >> cost;
// it is an object belongs to istream class ">>" refers to bits coming from
}
// Function to display food details
void display() {
cout << "!! FOOD ITEM LIST !!" << endl;
// it is an object belongs to ostream (output stream) class "<<" refers to bits are sending from memory to desktop

cout << "Name: " << name
<< " Color: " << color
<< " Taste: " << taste
<< " Cost: " << cost << endl;
// Outputs the food details in a structured format
}
};
```

```
int main() {
int n; // Variable to store the number of food items

cout << "Enter your food_limit: ";
// it is an object belongs to ostream (output stream) class "<<" refers to
bits are sending from memory to desktop
cin >> n;
// it is an object belongs to istream class ">>" refers to bits coming from

food temp; // Temporary object of class food to store and process

details

// Loop to input details of 'n' food items
for (int i = 0; i < n; i++) {
temp.set(); // Call the set function to input food details
}

// Loop to display details of 'n' food items
for (int i = 0; i < n; i++) {
temp.display(); // Call the display function to output food
details }

return 0; // Indicates successful execution of the
program }
==========================================================
==========================================================
```

==========

==========

#include <iostream>
// Include the iostream library for input/output operations

/*

Group_Fruits

Grapes, Watermelon, Kiwi, Mandarin,

Attributes:

- Color_fruit
- Count_items
- Size
- Texture
- Produced_in

Functions:

- No_of_Ripening_fruits()
- Aging_time_of_fruit()

*/

using namespace std; // Use the standard namespace to simplify code

// Define a class named 'fruits' to represent the properties and actions of fruits

class fruits {

string name, color, size, texture, produced_in; // Attributes of the fruit

```
int count_items; // Number of fruit items

public:
// Member function to input details of the fruit
void set() {
cout << "Enter name: "; // Prompt user for fruit name
cin >> name; // it is an object belongs to istream class">>" refers to
bits coming from
cout << "Enter color of it: "; // Prompt user for fruit color cin >> color; //
it is an object belongs to istream class">>" refers to bits coming from
cout << "Enter count: "; // Prompt user for count of items cin >>
count_items; // it is an object belongs to istream class">>" refers to bits
coming from
cout << "Enter size: "; // Prompt user for fruit size
cin >> size; // it is an object belongs to istream class">>" refers to
bits coming from
cout << "Enter texture: "; // Prompt user for fruit texture cin >> texture; //
it is an object belongs to istream class">>" refers to bits coming from
}

// Member function to display details of the fruit
void display() {
cout << "!! FOOD ITEM LIST !!" << endl; // it is an object belongs
to ostream (output stream)class"<<" refers to bits are sending from
memory to desktop
cout << "Name: " << name << " Color: " << color << " Texture: " <<
texture << " Count: " << count_items << endl;
// Display fruit details
```

```
}
};

int main() {
int n; // Variable to store the number of fruits
cout << "Enter your food_limit: "; // Prompt user to input the limit of
food items
cin >> n; // it is an object belongs to istream class">>" refers to
bits coming from

fruits temp; // Create an instance of the 'fruits' class
for (int i = 0; i < n; i++) { // Loop to input details of 'n' fruits
temp.set(); // Call the 'set' function to input fruit details
}
for (int i = 0; i < n; i++) { // Loop to display details of 'n' fruits
temp.display(); // Call the 'display' function to print fruit details
}

return 0; // Return 0 to indicate successful program
execution }
==========
=
==========
= ==========
```

```
==========
=
==========
```

= ==========

```
#include
using namespace std;

class vehicle {
string name, fuel_type, model, speed; // Member variables to
store vehicle information

public:
// Function to set the details of the vehicle
void set() {
// Prompt the user to enter the name of the vehicle
cout << "Enter the name of the vehicle: ";
cin >> name; // Accepting input for vehicle name
// Prompt the user to enter the fuel type of the vehicle
cout << "Enter the fuel type of " << name << " : ";
cin >> fuel_type; // Accepting input for fuel type

// Prompt the user to enter the model of the vehicle
cout << "Enter the model of " << name << " : ";
cin >> model; // Accepting input for vehicle model

// Prompt the user to enter the max speed of the vehicle
cout << "Enter the max speed of " << name << " : ";
cin >> speed; // Accepting input for max speed
}

// Function to display the max speed of the vehicle
void display_max_speed() {
```

```
// Display the max speed of the vehicle using cout (output stream)
cout << "Max speed of " << name << " is " << speed << "."; }
};

int main() {
vehicle a1; // Create an object of the 'vehicle' class
a1.set(); // Call the set function to get vehicle details
a1.display_max_speed(); // Call the display function to show the max
speed

return 0; // End of the main function
}

==========
=
==========
= ==========
13
==========
=
==========
= ==========
/* Group-Apps
Adobe Photoshop, Microsoft Word
Properties
Storage, Price, Boot_speed
Function
Compare_Boot_Speed()
*/
```

```
#include
using namespace std;

// Creating a class named "app"
class app {
string a; // Attribute to store app name (could be for example,
"Adobe Photoshop")
float storage, price, boot_speed; // Attributes to store storage, price, and
boot speed of apps

public: // Public access specifier to make functions accessible
from outside the class
// Function to take the app name as input
void name() {
cout << "Enter name of app" << endl; // Prompting user to enter
app name
cin >> a; // Reading app name from the user
// cout << "Enter name of app2" << endl;
// cin >> a2; // Not used in this version of the code
// cout << a1 << endl; // Not used in this version of the code
}

// Function to take properties (storage, price, and boot speed) of the app
as input
void properties() {
cout << "Enter the storage of " << a << " (in Mb): " <<
endl; cin >> storage; // Reading storage input for the app
cout << "Enter the price of " << a << endl;
```

```
cin >> price; // Reading price input for the app
cout << "Enter the Boot_speed (in secs) of " << a << " {ex:3}: " <<
endl; cin >> boot_speed; // Reading boot speed input for the app }

// Function to compare boot speed of two apps and print the result
void compare_boot_speed(app a1, app a2) {
if (a1.boot_speed < a2.boot_speed) // Compare boot speed of first
app with the second app
cout << a1.a << " is better, "; // Print the better app
else if (a1.boot_speed > a2.boot_speed)
cout << a2.a << " is better"; // Print the better app
else
cout << "Both are equal"; // If both have the same boot
speed }
};

// void compare_boot_speed() {
// // Placeholder for any future functionality if needed
// }

int main() {
app a1, a2, check; // Creating three instances of the app class

a1.name(); // Taking input for app1's name
a2.name(); // Taking input for app2's name

a1.properties(); // Taking input for properties (storage, price, boot speed)
of app1
a2.properties(); // Taking input for properties of app2
```

```
check.compare_boot_speed(a1, a2); // Comparing the boot speed
of app1 and app2 and displaying the result

return 0;
}

==========
=
==========
=
==========
14
==========
=
==========
= ==========
#include // Header file for input/output stream functions using
namespace std; // Allows the use of standard namespace without 'std::'

class student {
string name, clas, place; // Private members to store student's details
int roll, age;

public:
void set(); // Declaration of a function to set student details

// Member function to display student details
```

```
void display() {
cout << "name: " << name // Output student's name
<< " roll: " << roll // Output student's roll number
<< " class: " << clas // Output student's class
<< " age: " << age // Output student's age
<< " place: " << place << "\n"; // Output student's
place }
};

// Inline function for setting student details (efficient execution)
inline void student::set() { // Definition of the function to set
student
details
cout << "enter your name: "; // Prompt user to enter name cin >> name;
// 'cin' is an object of the 'istream' class, reading input from the user

cout << "enter roll: "; // Prompt user to enter roll
cin >> roll; // 'cin' is an object of the 'istream' class, reading input
from the user

cout << "enter class: "; // Prompt user to enter class
cin >> clas; // 'cin' is an object of the 'istream' class, reading input
from the user

cout << "enter age: "; // Prompt user to enter age
cin >> age; // 'cin' is an object of the 'istream' class, reading input
from the user
```

```
cout << "enter place: "; // Prompt user to enter place
cin >> place; // 'cin' is an object of the 'istream' class, reading input
from the user
}

int main() {
student s1; // Creating an object 's1' of the 'student' class
s1.set(); // Calling the 'set()' function to input student details

s1.display(); // Calling the 'display()' function to output the student details
return 0; // Indicating that the program has executed
successfully }

==========
=
==========
= ==========
```