```
#include <ctime>
#include <iostream>
#include <string>
using namespace std;
```
} header.

#include <ctime> C++ header for working with a time function

∴ Thir function used to generate a random number in this project.

#include <iostream> C++ header for working with Input & output functions.

∴ Thir function / header used to take a Input from the user and display output to the user.

#include <String> C++ header for working with a (Includes the string Class to work with strings).

∴ Thir ir used to String Variable to store the Character name.

Using namespace std ; allows the use of standard C++ library features like cout & cin. without std:: prefix

Q. Differnce Between #include<iostream> & using namespace std;

#include <iostream> is required to use the input & output function for all standard library.

using namespace std is optional but makes code by removing the need of prefix standard library names with std::

With using namespace std	without using namespace std.
#include<iostream> using namespace std; cout<<"Enter two NO"<< endl; cin>>a; cin>>b;	#include<iostream> std::cout<<"Enter two NO"<< endl; std::cin>>a; std::cin>>b;

In class character

protected

Symbols public :- +
protected :- #
private :- -

protected :- Inside class : where members can be accessed normally by all members of the class.

Derived class :- The member's can be accessed by classes that inherit from the class the member is defined (Base class).

outside class :- No The members cannot be accessed directly by non-member function or object. Unless they are friends of the class.

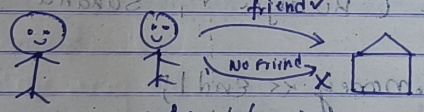
Q. What is Friend Function ?

Friend Function that is given special access to the private & protected member of a class even though it's not a member of that class.

A function as a friend by using the friend keyword inside the class definition.

Syntax :-

Friend return-type Functionname (class name & reference);



My Asha is my friend She can access my home resource.

By code Example :-

```
#include<iostream>
using namespace std;
class Sahana;

Class Rutuja {
private:
int money = 20;
friend Void Suraksha (Rutuja, Sahana) ;
};

Class Sahana {
private:
int money = 10;
friend void Suraksha (Rutuja, Sahana);
};

Void Suraksha ( Rutuja r, Sahana s)
{
    cout << r.money << endl;
    cout << s.money << endl;

    cout << "Sum" << r.money + s.money <<
    endl;
}

int main ()
{
    Sahana s;
    Rutuja r;
    Suraksha (r, s);
    return 0;
}
```

string name;
int health;
int shield;

public:

Character (const string &NAME, int HEALTH
int SHIELD) : name (NAME), health (
HEALTH), shield (SHIELD)
{ }

Destructor :- Special member function that is called when an object is destoryed.

Virtual destructor :- Virtual destructor is a destructor that is declared with the Virtual keyword. (Inheritance).

Virtual destructor is declared with the Virtual keyword.

It Ensure that when a derived class object is deleted through a Base Class pointer.

The destructor of the Base class is called followed (then) then the destructor of base class.

Q Why Virtual destructor ? A Virtual destructor ensure that all resource allocated to both both Base and derived classes are properly released, avoid memory leaks.

```
String GetName() const {
    return name;
}
```

member function does not modify the state of the object on which its called.

pure Virtual Function :- Is a function that is declared in a base class but not have to impument in that class.

It is meant to overidden by derived classes.

Declearing a function as pure virtual is way to create an abstract class.

Syntax :-

Virtual returntype functionName() = 0;

Q What is Abstract class?

A class that contains at least one pure virtual function is called an abstract class.

Virtual Function :- It's a member function in a base class that you expect to override in derived classes.

When you are declearing member function as virtual, you are informing the compiler that the function can be overridden and that it should use dynamic binding to determine which function to call based on actual type of object, not the type of the pointer or reference.

Virtual Function	Pure Virtual Function.
Has to be Implemented in the base class.	Does not have an Implementation in the base class.
Allows derived classes to override and provide new Implementation.	Defines an Interface that derived classes must implement.

```
Void GetAttacked()
{
    int roll = rand()%100;   // to generate a random integer.
    if (roll == 0)
    {
        health = 0;
    }
    else if (roll <= 50)
    {
        shield = shield - (2 - (rand()%4));
                          (0 1 2 3)
        if (shield < 0)
        {
            health = health + shield;
            shield = 0;
        }
    }
    else if (roll <= 79)
    {
        health = health - (4 + rand()%5));
                         (0,1,2,3,4)
    }
}
```

4 + (0%5) = 0 = 4
4 + (1%5) = 0 = 5
4 + (2%5) = 2 = 6
4 + (3%5) = 3 = 7
4 + (4%5) = 4 = 8

int roll = rand() % 100. Divide by 100.
which results on a value 0 and 99.

These values are stored in variable roll.

if (roll == 0)
{ health = 0; } The character was hit in a special way.

else if (roll <= 50)
This condition checks if roll is between 1 and 50
This represents a range where the character takes damage but not severe as a critical hit.

Shield = Shield - (2 - (rand() % 4));

This line modifies the Shield value.
rand % 4 gives a random number 0 3

0 1 2 3

2 - (rand() % 4)

2 - 0 = 2
2 - 1 = 1
2 - 2 = 0
2 - 3 = -1

if (Shield < 0)
{
health = health + Shield;
Shield = 0;
}

51 52 53 79

```
void getAttacked () override
int roll = rand () % 100;
if (roll < 15)
{
if (++shield > maxShield)
{
  shield = maxShield;
}
--shield;
}
roll = rand % 100;
if (roll < 65)
{
shield = shield - (1 + rand () % 2));
if (shield < 0)
{
  health = health + shield;
  shield = 0;
}
}
shield = -5
health = 20
20 + (-5)
20 - 5 = 15
```