Graphical user interface

Description automatically generated

**Association(By value):** I have a relationship with an object. Foo uses Bar.

class Bar {};

class Foo {

void fun (Bar b)

};

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

**Aggregation(By reference):** Type of association. I have an object which I have borrowed from someone else.

Independent. One object will carry reference of another object. One object destroy, other will not destroy. weak relationship.

**Example: Person object has reference of Car object. If person object is destroyed, Car object will not be destroyed.**

class Car

{ int model;

string name;

public:

Car(string name, int model): name(name), model(model){}

void printCarInfo()

{

cout<<model<<name<<endl;

}

};

class Person

{

string name;

Car \*mycar;

public:

Person() = default;

Person(string name, Car\* mycar):name(name), mycar(mycar)

{

}

};

int main()

{

Car c ("BMW", 134);

Person \*p = new Person("Ankit", &c);

delete p;

c.printCarInfo();

return 0;

}

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

**Composition(composed inside other class):** Type of association. Dependent. One object will carry another object as a value. One object destroy, other will also destroy. Strong relationship.

**Example: Car object has Engine object. If Car object is destroyed, Engine object will be destroyed.**

#include <iostream>

#include <string>

using namespace std;

class Engine

{

int power;

public:

Engine(int power):power(power)

{

cout<<"Engine object is created"<<endl;

}

~Engine()

{

cout<<"Engine object is destroyed"<<endl;

}

};

class Car

{

int model;

string name;

Engine eng;

public:

Car(string name, int model, Engine eng): name(name), model(model), eng(eng)

{

cout<<"Car object is created"<<endl;

}

~Car()

{

cout<<"Car object is destroyed"<<endl;

}

};

int main()

{

Engine e(10);

Car \*c = new Car ("BMW", 134, e);

delete c;

return 0;

}

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

Composition is has-a relationship while Inheritance is is-a relationship.