<https://www.youtube.com/watch?v=Eaz0P_gJ9FE>

<https://www.hackerearth.com/practice/notes/virtual-function-and-virtual-table-in-c/>

Virtual Function and Virtual Table in C++

Runtime Polymorphism is C++ is achived with virtual function. Hence in order to implement the runtime behaviour C++ compiler inserts Virtual Table for every class having virtual function or class inherited from the class that has virtual function.

Following post explains virtual table entry and runtime behaviour.

Consider following code

#include<iostream>

using namespace std;

class Vehicle

{

public:

virtual void ApplyBreaks()

{

cout<<"Vehicle Break"<<endl;

}

virtual void ApplyHorns()

{

cout<<"Vehicle Horns"<<endl;

}

};

class Truck: public Vehicle

{

public:

void ApplyBreaks()

{

cout<<"Truck Break"<<endl;

}

};

class Car:public Vehicle

{

public:

void ApplyHorns()

{

cout<<"Car Horns"<<endl;

}

};

Above is a classic example of classification inheritance.

If we print sizeof Vehicle class it will give us 4 bytes. As the class contains a virtual function C++ compiler inserts a pointer v\_ptr called as virtual table pointer. Compiler also creates a table for that class called as virtual table known as vtable for the class. The table is created compile time v\_ptr holds the address of v\_table of the corresponding class. vtable is a array of function pointers pointing to virtual function.

virtual table contains pointers to ApplyHorn function and ApplyHorns of Vehicle class. Hence

Vehicle \*v = new Vehicle();

v->ApplyBreak(); //Calls vehicle ApplyBreak

v->ApplyHorn(); //Calls vehicle ApplyHorn

As Truck Class inherits Vehicle class, and as we know Vehicle class has a data member v\_ptr. Truck class inheits v\_ptr of Vehicle class but new Virtual table will be created compile time for Truck class. hence v\_ptr of Truck class holds the address of vtable of Truck class. As Truck class has implemented ApplyBreak and ApplyHorn implementation is missing. vtable of Truck class contains pointer to ApplyBreak function of Truck class but pointer to ApplyHorn function of base class i.e Vehicle class.

Vehicle \*v = new Truck();

v->ApplyBreak(); //calls Truck ApplyBreak

v->ApplyHorn(); //calls Vehicle ApplyHorn

Similarly goes with Car class.

Vehicle \*v = new Car();

v->ApplyBreak(); //calls Vehicle ApplyBreak

v->ApplyHorn(); //calls Car ApplyHorn