Jhf5my - John Fishbein Wednesday, April 10, 2019

CS 2150 In-Lab 9 Report

Dynamic Dispatch

Dynamic Dispatch is the situation in which it is unclear at compile time which class member function should be invoked in terms of inheritance. When this occurs in c++, if the virtual keyword is present in the method header, the decision is made at run time, otherwise the compiler makes a decision at compile time. When the virtual keyword is present, the run-time decision is made using a virtual method table. Essentially, when the object is instantiated the address of the desired method is placed into the virtual method table. A pointer is then added to the object which points to the table, allowing it to be accessed. This allows the proper method to be called at runtime. Consider this example code below: If the virtual keyword were not in front of func, the compiler would not know which version of func to go to when var->func is invoked. Because it would not know, the c++ compiler would default to the func method in class A. When executed without the virtual keyword, this code prints 0. But, the virtual keyword is present. This causes a virtual method table to be created when var is instantiated. Because of this table, the compiler is able to figure out which version of func to go to and therefore would execute the func in class B. When the virtual keyword is present, this code prints 10.

```
class A {
public:
  int x;
  A(){ x =0;}
  A(int n){
    x = n;
  }

  virtual int func(){
    return x;
  }
};
class B : public A {
public:
  int y;
  B(){y = 0;}
  B(int n){
    y = n;
  }

  int func(){
    return 2*y;
  }

};
```

```
int main(){
  A* var = new B(5);

  cout << var->func() << endl;


  return 0;
}
```