Amber Liu
Al7bf
11/20/2018
inlab9.pdf

Dynamic dispatch is when the program would determine which method to call at runtime. This means that the computer does not know until after the code is ran after compiling. The program would pass the virtual method by calling from the address, instead of method's name directly. The virtual keyword is important because it is the only way dynamic dispatch will work when it comes to main code and how it will run.

The code I wrote:

```
#include<iostream>
#include<string>

using namespace std;

class person{
public:
    virtual void name(){}
};

class student: public person{
    virtual void name(){}
};

int main(){
    person *x;
    int num = 0;
    if(num){
        x = new student();
    }
    else{
        x = new person();
    }
    x->name();
    return 0;
}
```

In the code, I have class called person and subclass that inherits the person class called student and both contain the function name. The dynamic dispatch is used through the virtual keyword, and there is a pointer of person type that remains unknown until after the actual code is ran after compiling. The pointee is set first, then check and run overridden one instead. This is where the decision on which function to call is made by using run-time type of an object.