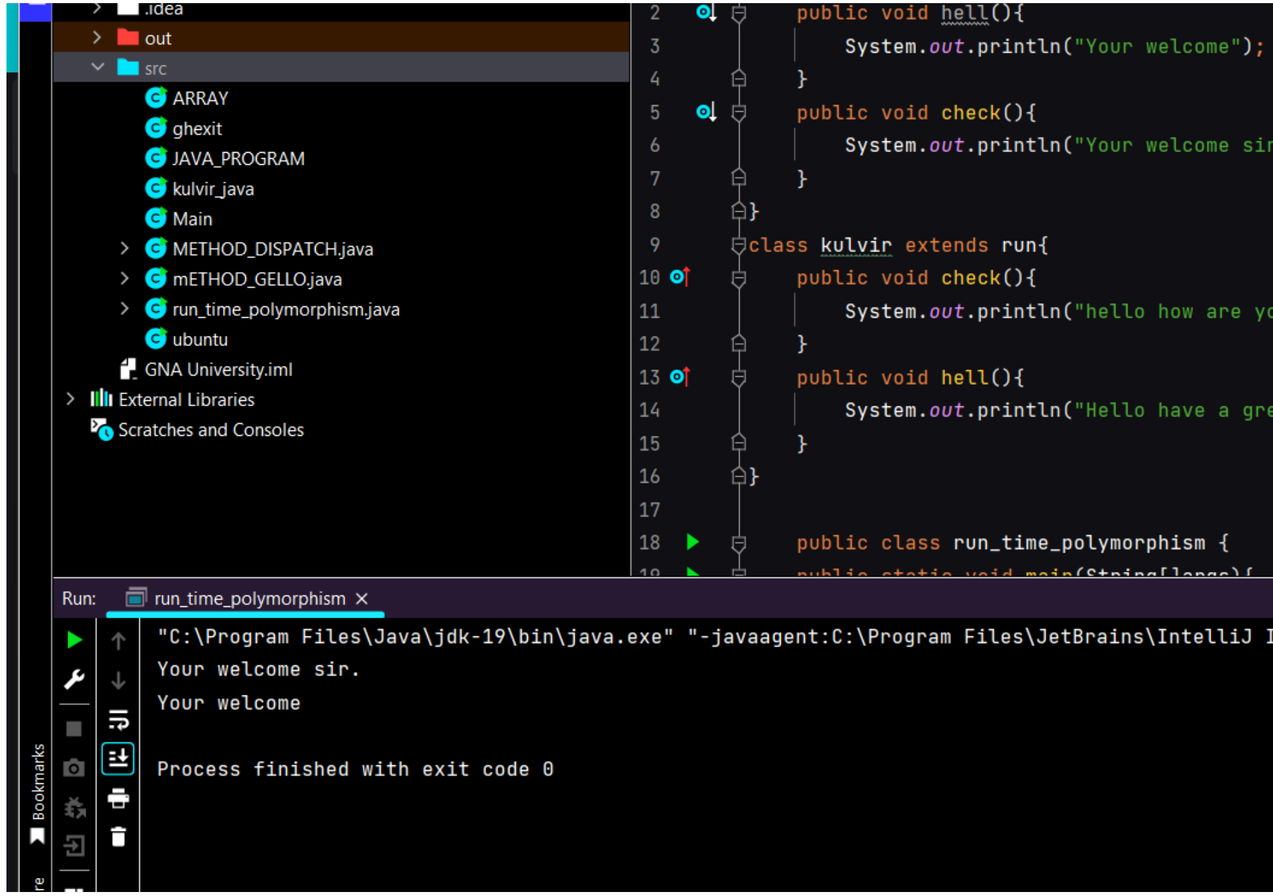
GNA University

Dynamic Method Dispatch

-By Kulvir Singh Saggu
(GU-2021-4246)

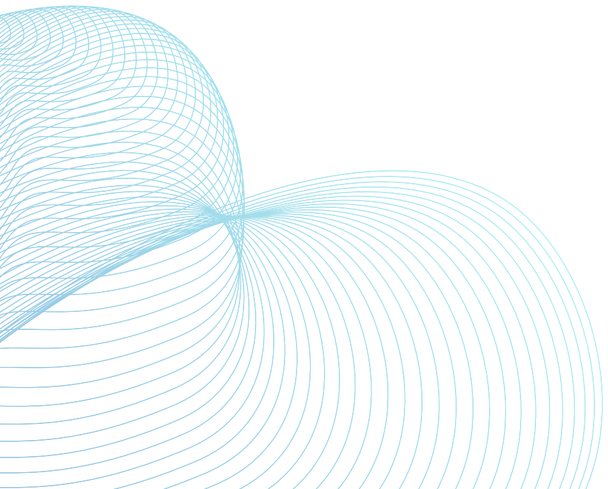
AGENDA

- Dynamic Method dispatch
- Syntax and code
- How the code works
- Upcasting
- Points To Remember



DYNAMIC METHOD DISPATCH

The Basic Points we have to know before we go further because dynamic method dispatch follow these points as you can see.



1. **Using Method Overriding we can achieve runtime polymorphism.**
2. **Dynamic Metod Dispatch is a Mechanism that decides which overridden method will be called at runtime.**
3. **Here we use the refererence of Superclass and object(instance) of Subclass.**
4. **Based on the object being referred, it will decided which method will be called.**

SYNTAX

- Superclass obj = new Subclass_obj( )

- **Note:- We can use reference variable of superclass and obj of subclass.**
- **It is worked on run time polymorphism,we can say it is dynamic binding or late binding.**

Code:-

We know that run time polymorphism work on method overriding, having same fuctions and same parameters.

```
class Animal {
    void breath() {
        System.out.println("Breathing...");
    }
}

class Dog extends Animal {
    void breath() {
        System.out.println("Breathing as Dog...");
    }
}

class Fish extends Animal {
    void breath() {
        System.out.println("Breathing as Fish...");
    }
}
```

In the main class we write:-

We know that run time polymorphism work on method overring, having same fuctions and same parameters.

```
class methodOverrideDemo {
    public static void main(String args[]) {
        Animal a;
        a = new Dog();
        a.breath();
        a = new Fish();
        a.breath();
    }
}
```

OutPut:

```
1  Breathing as Dog...
2  Breathing as Fish...
```

Run-time polymorphism cannot be achieved by data members

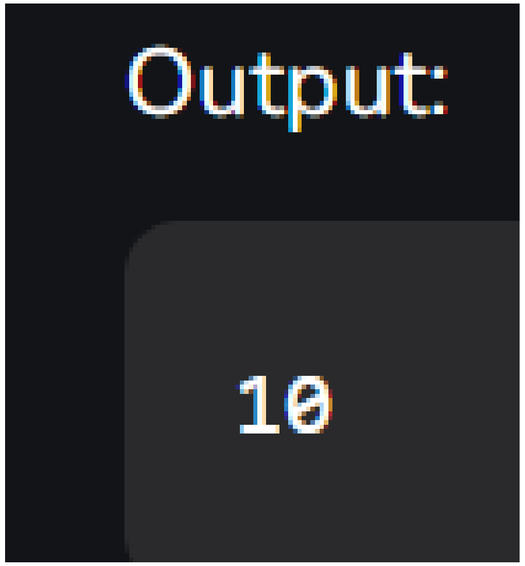
```java
// Java program to illustrate the fact that
// runtime polymorphism cannot be achieved
// by data members

// class A
class A
{
    int x = 10;
}

// class B
class B extends A
{
    int x = 20;
}

// Driver class
public class Test
{
    public static void main(String args[])
    {
        A a = new B(); // object of type B

        // Data member of class A will be accessed
        System.out.println(a.x);
    }
}
```



Explanation of flow of execution of the above program:

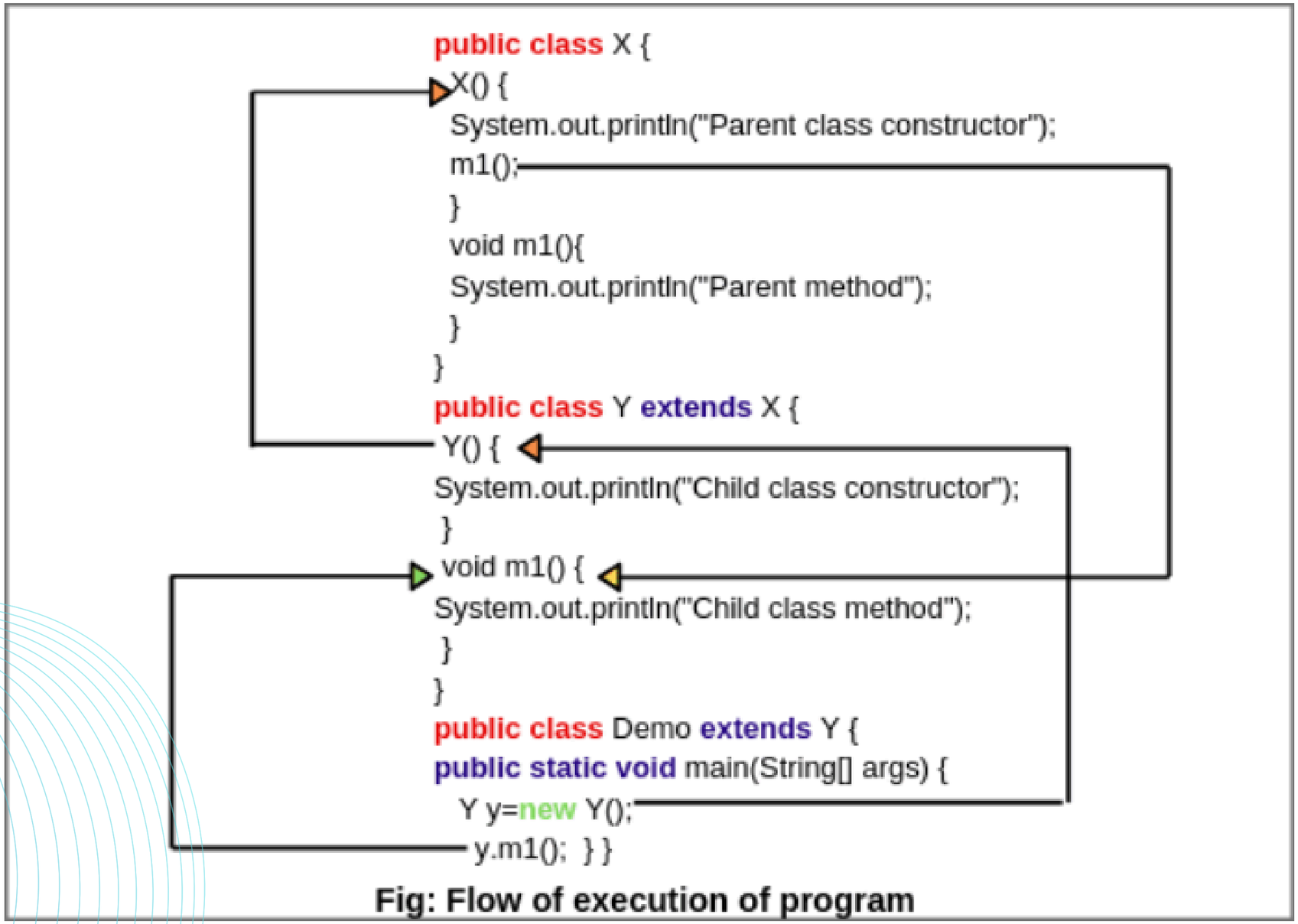


Fig: Flow of execution of program

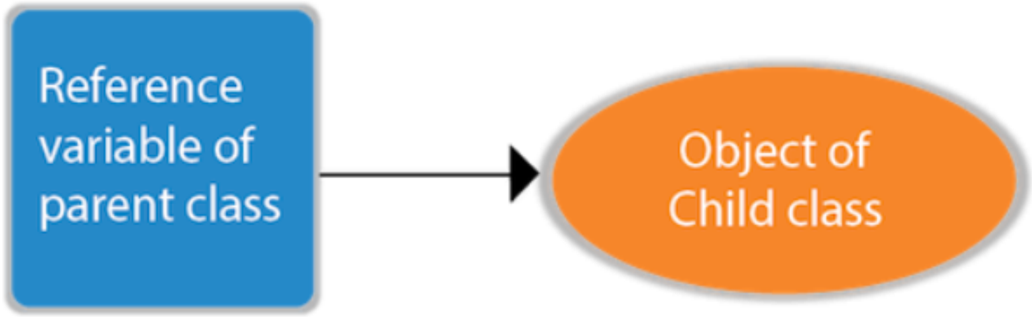
How the superclass and subclass will assigned?

```
class Bank {
    int maximumInterestRate = 15;
}

class Sbi extends Bank {
    int maximumInterestRate = 9;
}

public class MethodOverridingDemo {

    public static void main(String[] args) {
        Bank b = new Sbi();
        System.out.println("Maximum Interest Rate : " + b.maximumInterestRate);
    }
}
```



Upcasting

It means that we can access overriden methods of a child class or subclass.We cannot acces their attributes and variables.

- **Here we will doing upcasting:-**
- **For example:-**

  Parent a = new child();

[Here parent reerence variable will access the child method of a subclass.

Points to remember

Dynamic Method Dispatch in Java

- Dynamic method dispatch is also known as run time polymorphism.
- It is the process through which a call to an overridden method is resolved at runtime.
- This technique is used to resolve a call to an overridden method at runtime rather than compile time.
- To properly understand Dynamic method dispatch in Java, it is important to understand the concept of upcasting because dynamic method dispatch is based on upcasting.

THANKYOU