hasNextDouble
hasNextInt

```
if(sc.hasNextDouble())
{

}
```

OOP

System
{
InputStream in
PrintStream out
}

javac Program.java

Program.class
Employee.class

java Employee

```
class Program{

}

class Employee{

main()
}
```

src
Program.java

bin
Program.class

```
javac -d ..\bin Program.java
SET CLASSPATH=..\bin
java Program
```

```
Time t1 = new Time();
t1.
```

MethodArea
Java Stack -> t1
Java Heap -> new Time()

```
Scanner sc = new Scanner(System.in);
sc.nextDouble()
```

Naming Convention
Camel Case
- Every first letter of the word except first word should be made as capital
- it should be used for variables, fields, methods method parameters

Pascal Case
- First Letter of every word should be capital
- class, interfaces, enums

```
totalSalary;
calculateTotalTax(){
}
```

Employee

AttendanceSystem

NullPointerException
Scanner
Printstream

Datatypes
- It defines 3 things
1. Nature
   - What type of data can be stored inside it.
2. Memory
   - Memory required to store the given type of data
3. Operations
   - Operations that can be carried out on the given type of data

1. Primitive Datatypes (Value type)
   a. Boolean - boolean
   b. Character - char
   c. Integral - byte, short, int, long
   d. Floating-point - float, double
2. Non Primitive Datatypes (Reference type)
   a. Array
   b. Class
   c. Interface
   d. Enum

Literals
1. Boolean Literal
2. Character Literal
3. Integral Literal
4. Floating Point Literal
5. String Literal
6. Null Literal

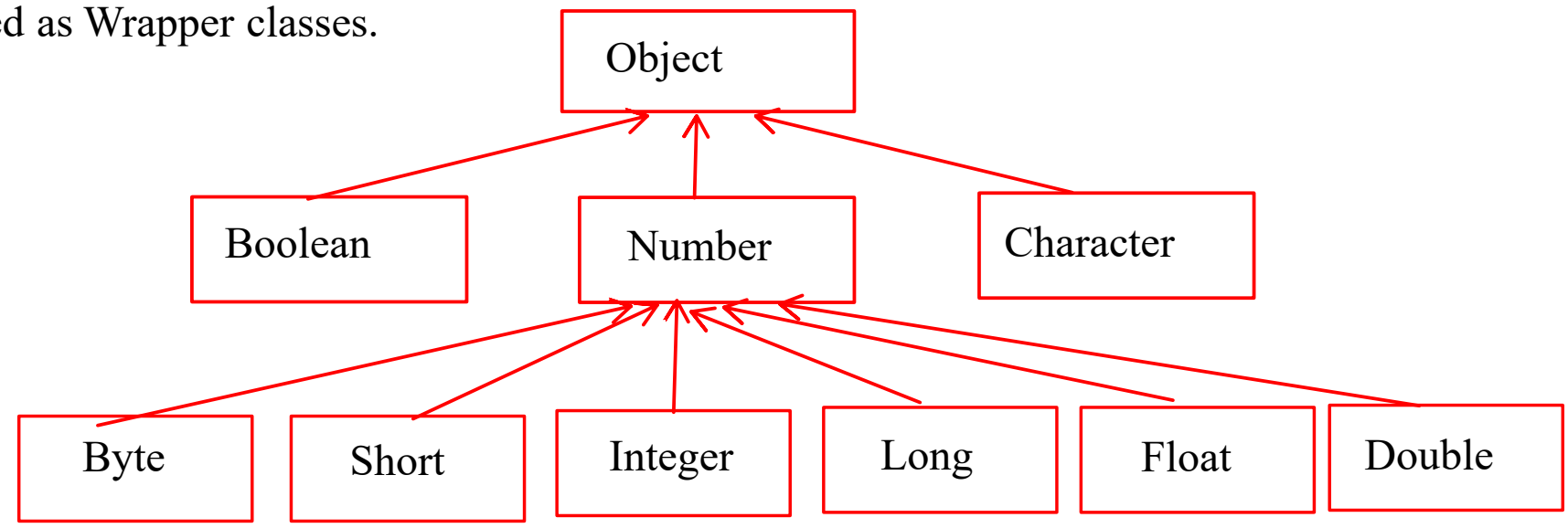
Widening
- Converting the narrower type of data into wider type is called as widening
- No explicit typecasting is required

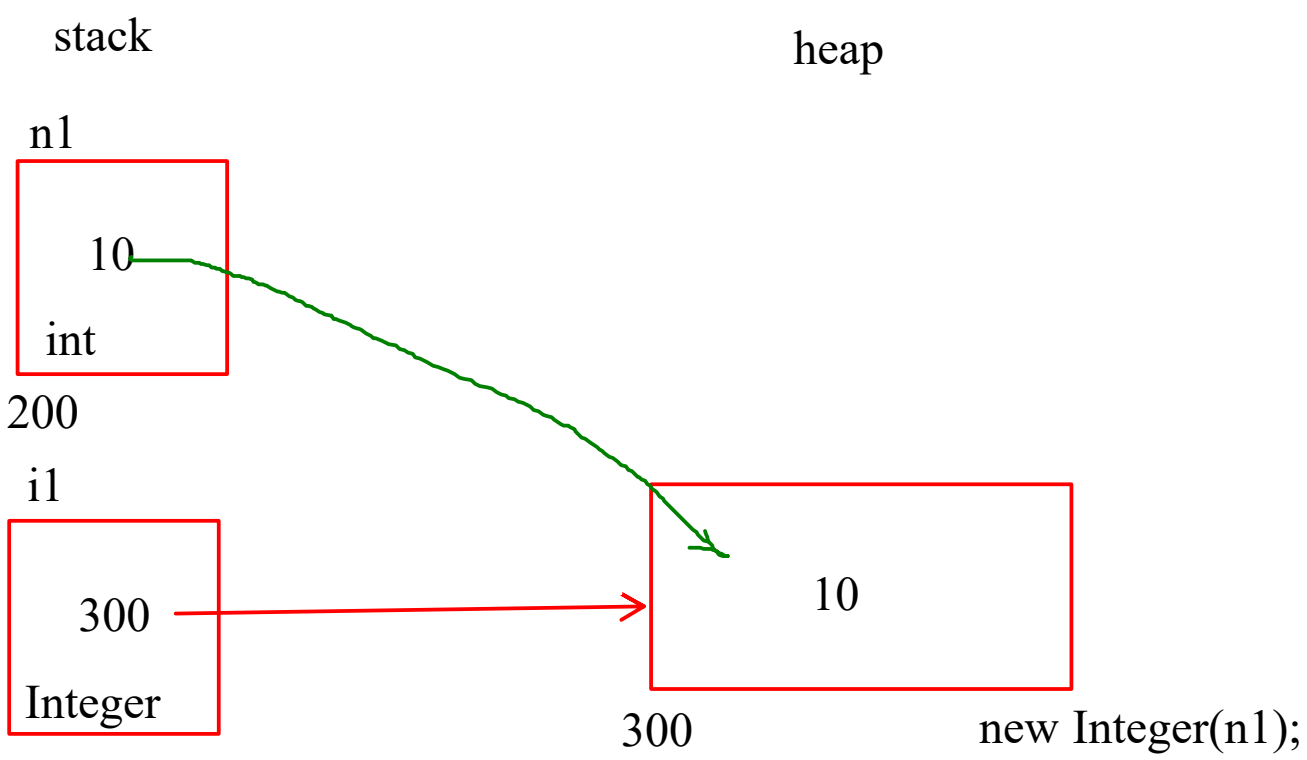
Narrowing
- Converting the wider type of data into narrower type is called as narrowing
- explicit typecasting is required

- We cannot convert Boolean type to any other type.
- Conversion of Character type to Intergral or Floating-point type is called as type conversion

Wrapper class
- For every primitive type, java have provided a class for it.
- These classes are called as Wrapper classes.

boolean -> Boolean
char -> Character
byte -> Byte
short -> Short
int -> Integer
long -> Long
float -> Float
double -> Double

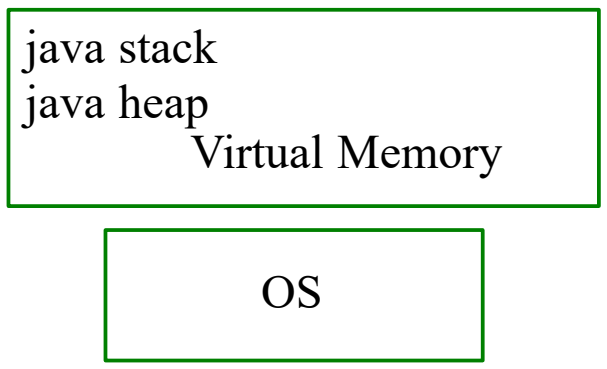


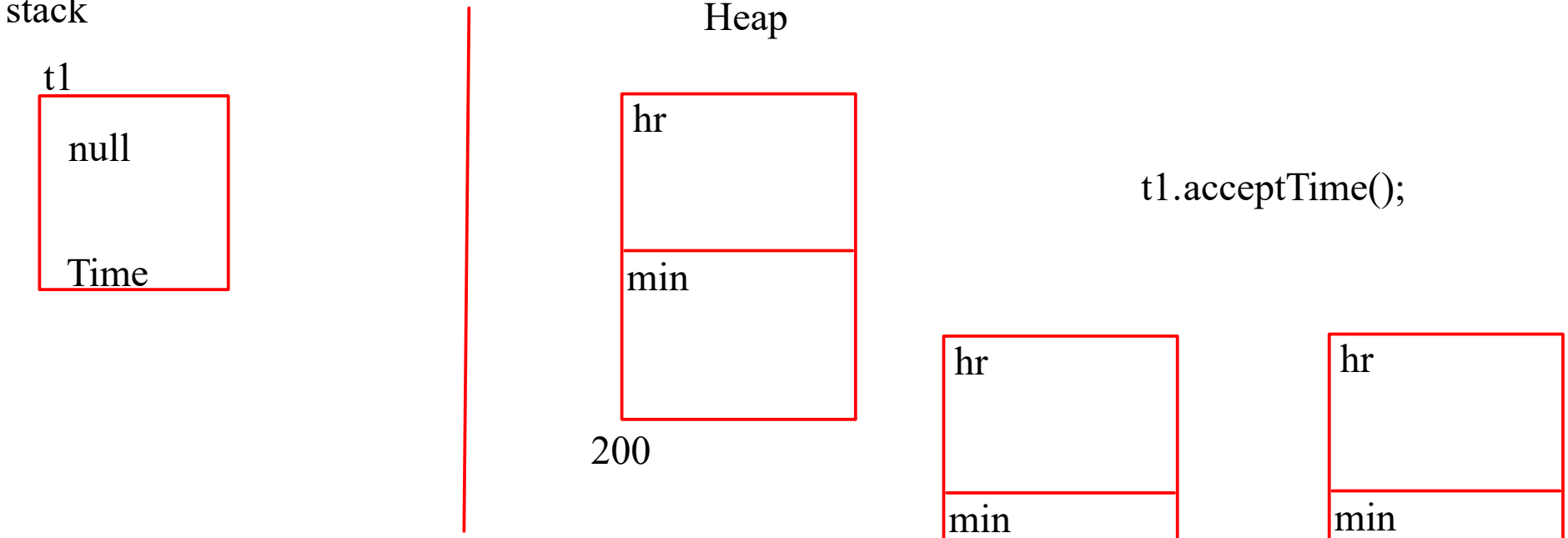


Time t1;
Time t2(120);
Time t3;
t3 = 120;

operator int(){
}

Time=int





Base *bptr;
bptr->f1();
bptr->f2();

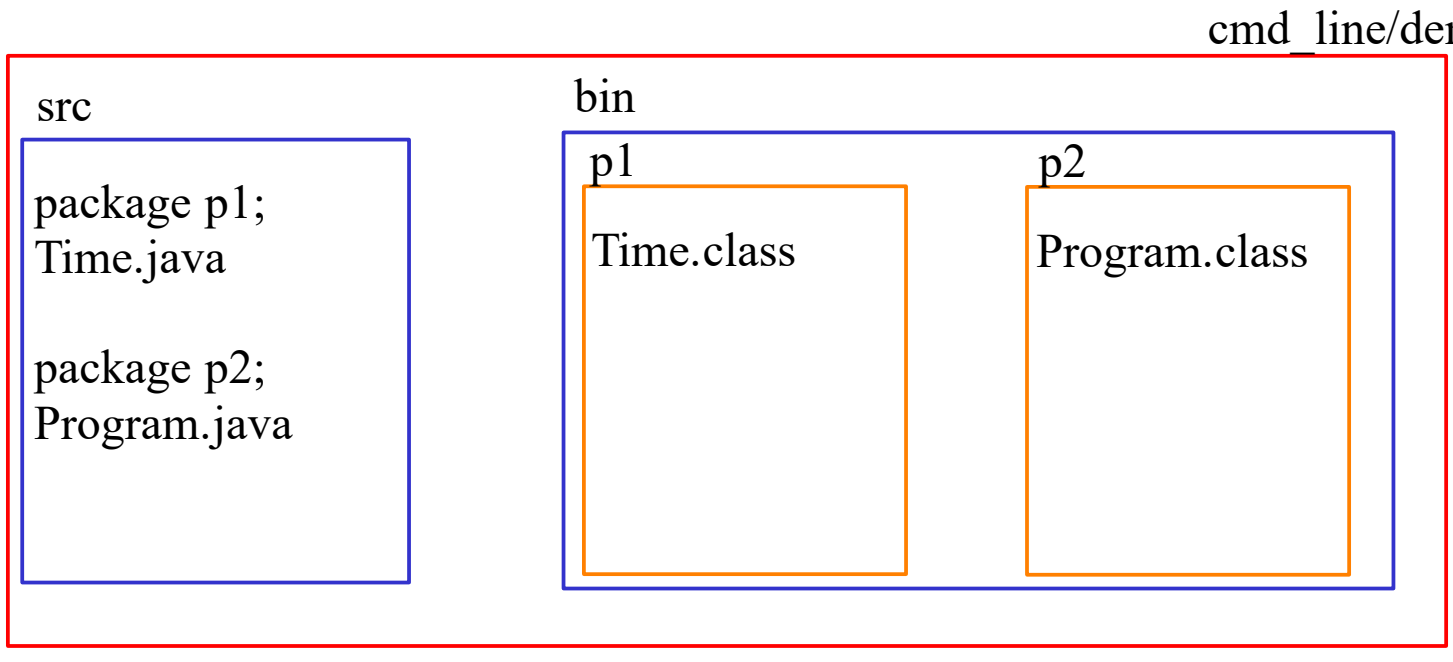
Packages

package p1
Time.java

package p2
Program.java

namespace NEmployee{
class Employee{
}

void findemployee();
void loadEmployees();
void saveEmployees();
}



// compile from src;
javac -d ..\bin Time.java
SET CLASSPATH=..\bin
javac -d ..\bin Program.java
java p2.Program

sunbeam.com

com.sunbeam.demo.entity
com.sunbeam.demo.tester

12345_D1_Rohan

Access Modifiers (visibility)
private
default
protected
public

Lab
1. Revise the concepts
2. Create a project in STS along with package
3. menu driven code in java
4. Revision on Types of

com.sunbeam.taxapp.entity
com.sunbeam.taxapp.tester
com.sunbeam.taxapp.constants

com.swiggy.foodordering

com.zomato.foodordering
com.zomato.egrocery

String packagename = "com.swiggy.foodordering";

com.infosys.BlinkIT

com.sunbeam.BlinkIT