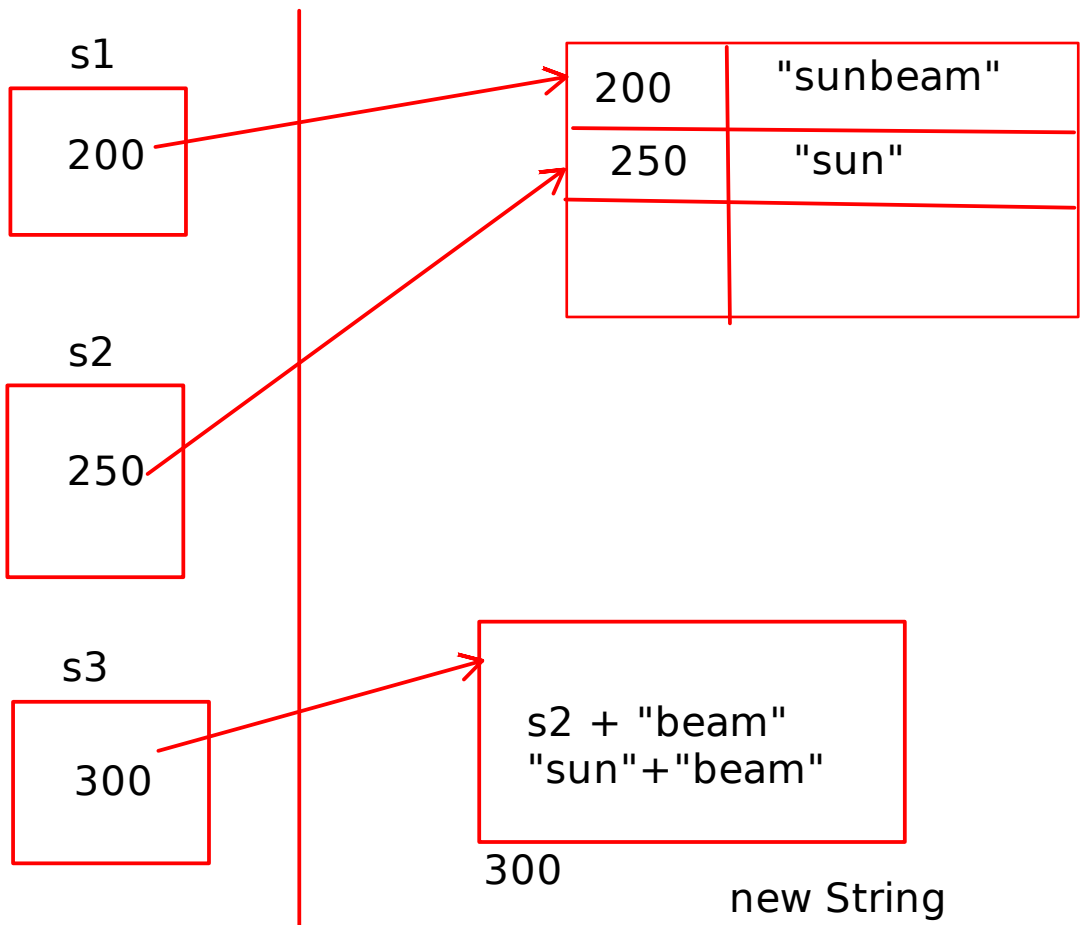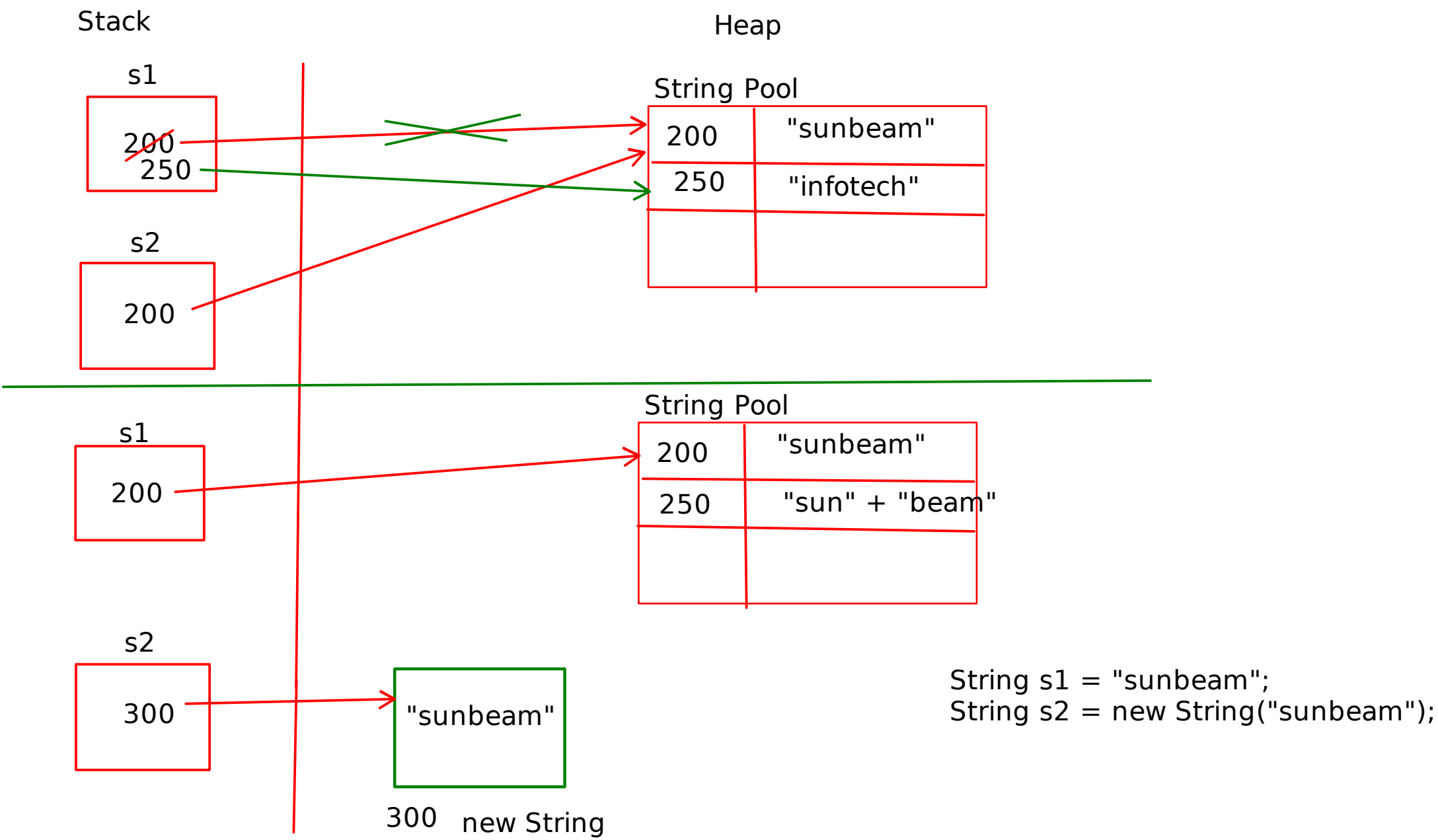Exception
Problems that occurs when program is in execution
1. Errors
    - Dont Handle
2. Exceptions
    - Mandatory to handle
    - 1. Checked
    - 2. Unchecked

String
- It is a class in java,
- java.lang

String s1 = "sunbeam";
String s2 = "sunbeam";

throw, throws

```
try{

}
catch(Exception e)// Generic catch block
{

}
```

s1==s2

String is immutable
s1 = "infotech";

Stack   Heap

s1
| 200 |
| 250 |

String Pool
| 200 | "sunbeam" |
| 250 | "infotech" |

s2
| 200 |

String Pool
| 200 | "sunbeam" |
| 250 | "sun" + "beam" |

s1
| 200 |

s2
| 300 | → "sunbeam"

300   new String

String s1 = "sunbeam";
String s2 = new String("sunbeam");

s1
| 200 |

| 200 | "sunbeam" |
| 250 | "sun" |

s2
| 250 |

String s1 = "sunbeam";
String s2 = "sun";
String s3 = s2 + "beam";

s3
| 300 | → s2 + "beam"
         "sun"+"beam"

300

new String

javac Program.java
java Program

Enum

```
enum ArithmeticOperations{
ADD=1,SUBSTRACT,MUlTIPLY
}
```

menu driven code

```
int choice= 2;
switch(ArithmeticOperation(choice)){
case ADD:

case SUBSTRACT:

}
```
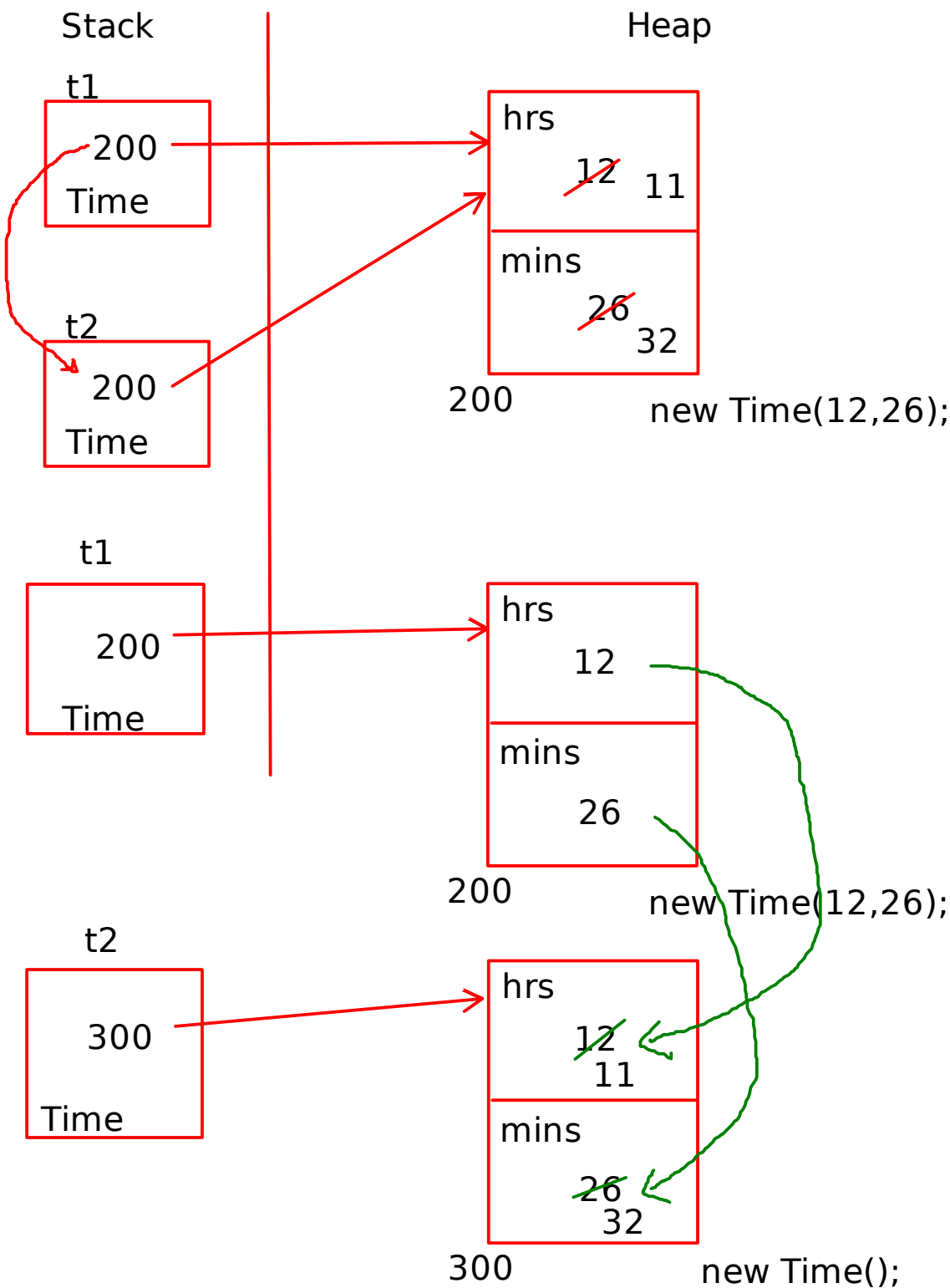
```
class Arithmeticoperation{
//datatype              identifier;
PSF ArithmeticOperations EXIT; // reference
PSF ArithmeticOperations ADD; // reference
PSF ArithmeticOperations SUB; // reference
PSF ArithmeticOperations MUL; // reference
PSF ArithmeticOperations DIV; // reference

private ArithmeticOperations(String name, int ordinal){
      super(name,ordinal); // sole constructor
}

static{
      ADD = new ArithmeticOperation("ADD",1);
      SUB = new ArithmeticOperation("SUB",2);
      .
      .
      $values[] = {EXIT,ADD,SUB,MUL}
}
}
```

Stack   Heap

t1

200
Time

t2

200
Time

hrs
    12 11

mins
    26
       32

200   new Time(12,26);

t1

200

Time

hrs
       12

mins
       26

200   new Time(12,26);

Time t2 = t1; // reference copy

Time t2 = t1.clone(); // Object copy

t2

300

Time

hrs
    12
       11

mins
    26
       32

300   new Time();

```
class Car{
String name;
String number;
Wheel wheel = new Wheel();
}
```

```
class Wheel{
double diameter;
double width;
}
```

Car has-a Wheel;

Car c1 = new Car();

Car c2 = c1.clone();

c1

| 200 |
| Car |

name
"car1"

number
"MH1234"

wheel
300

200

new Wheel();

diameter
20

width
300

300

c2

| 400 |
| Car |

name
""

number
""

wheel
500

400

diameter
0

width
0

500

new Wheel();
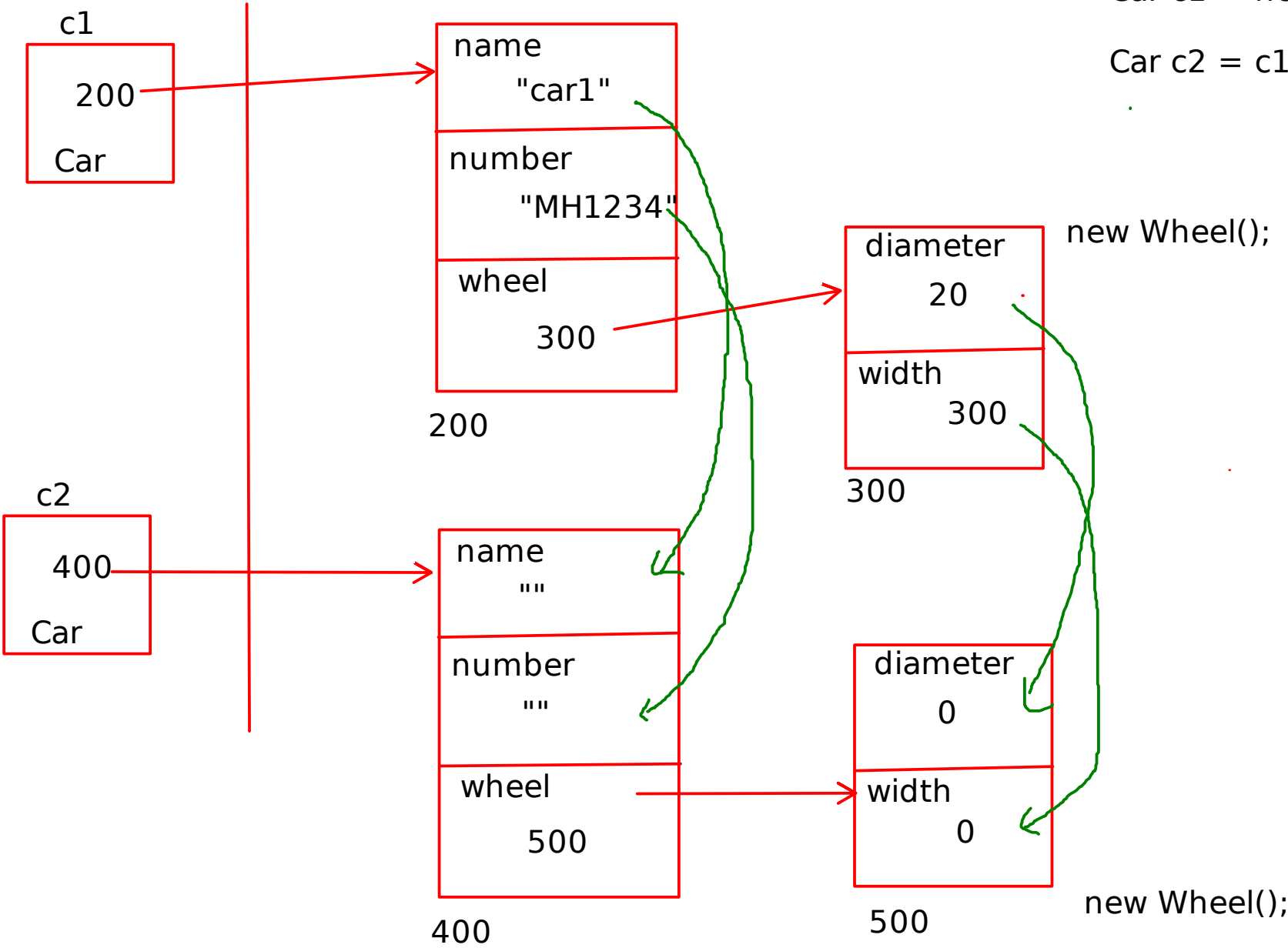
```
class Test{
int num1;
int num2;
}
```

```
class Test{
int *num1Ptr;
int *num2Ptr;
}

Test t1(10,20);

Test t2 = t1;
```

Default copy Ctor -> Object class clone()
User defined Copy Ctor -> User defined clone ()

t1

*num1Ptr
200

*num1Ptr
300

10

200

new int(10);

10

300

new int(20);

t2

*num1Ptr
200

*num1Ptr
300