1、关于变量的赋值：

1、如果变量是基本数据类型，此时赋值的变量所保存的是数据值。

2、如果变量是基本数据类型，此时赋值的变量所保存的是地址值。

```java
class Order{
 int orderId;
}
public class ValueTransferTest {
 public static void main(String[] args) {
 System.out.println("======基本数据类型======");
 int m = 10;
 int n = m;
 // 赋值的变量所保存的是数据值
  System.out.println("m = " + m + ", n = " + n); // m = 10, n = 10
  m = 20;
  System.out.println("m = " + m + ", n = " + n); // m = 20, n = 10

  System.out.println("======引用数据类型======");

  //赋值的变量所保存的是地址值
  Order o1 = new Order();
  o1.orderId = 1001;
  Order o2 = o1;
  System.out.println("o1 = "+ o1.orderId + ", o2 =" + o2.orderId); // o1 = 1001, o2 =1001
  o2.orderId = 1002;
  System.out.println("o1 = "+ o1.orderId + ", o2 =" + o2.orderId); // o1 = 1002, o2 =1002
  }
 }
```

2、方法形参的值传递机制：值传递

1、形参和实参

形参：方法定义时，声明在小括号里面的参数

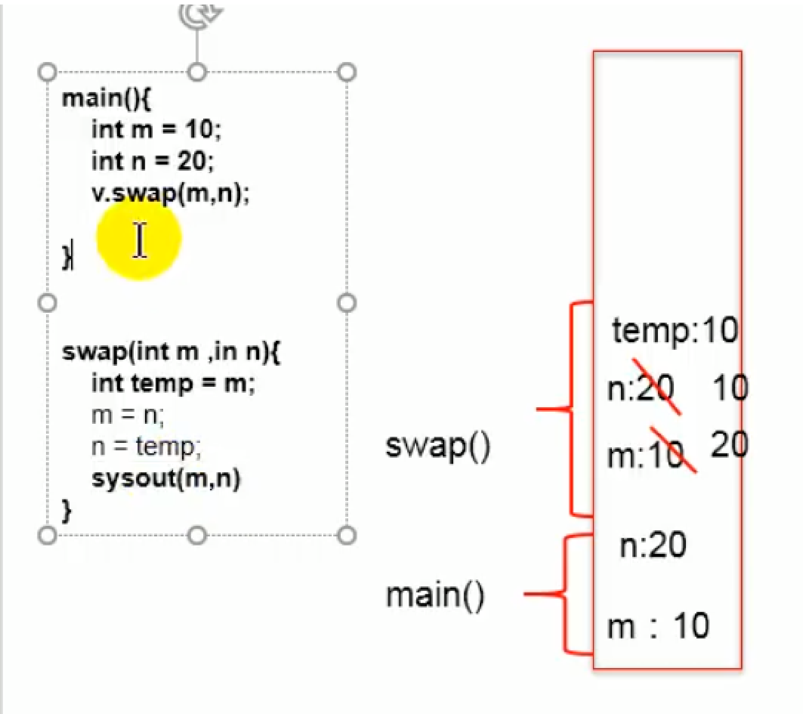
实参：方法调用时 实际传给形参的参数值

2、值传递机制

1、形参是基本数据类型

将实参基本数据类型变量的“数据值”传递给形参

```
public class ValueTransferTest2 {
 public static void main(String[] args) {
 int m = 10;
 int n = 20;
 System.out.println("m = " + m + ", n = " + n);

 //交换两个变量的操作
 int tmp = m;
 m = n;
  n = tmp;
  System.out.println("m = " + m + ", n = " + n);
  }
 }
```



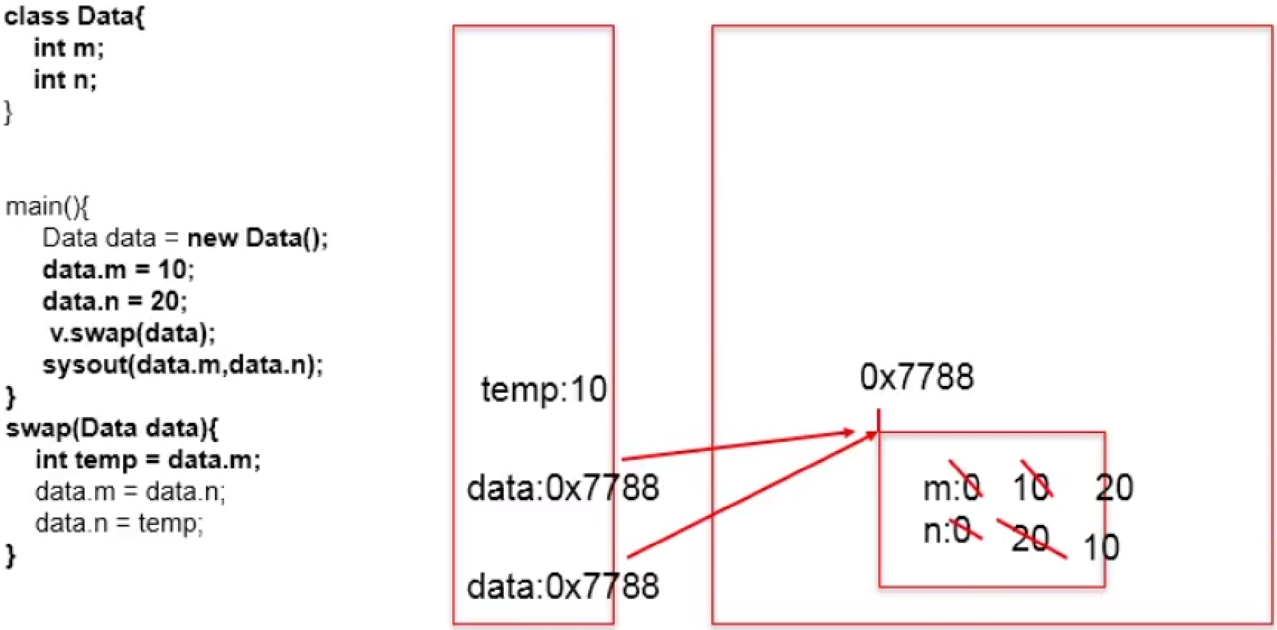
```
main(){
   int m = 10;
   int n = 20;
   v.swap(m,n);

}

swap(int m ,in n){
   int temp = m;
   m = n;
   n = temp;
   sysout(m,n)
}
```

2、形参是引用数据类型

将实参引用数据类型变量的“地址值”传递给形参

例题1：

```
class Data{
 int m;
 int n;
}
public class ValueTransferTest1 {
 public static void main(String[] args) {
 Data data = new Data();
 data.m = 10;
 data.n = 20;
  System.out.println("m = " + data.m + ", n = " + data.n);

  ValueTransferTest1 v = new ValueTransferTest1();
  v.swap(data);
  System.out.println("m = " + data.m + ", n = " + data.n);

  }
public void swap(Data data){
int tmp = data.m;
data.m = data.n;
data.n = tmp;
}
}
```

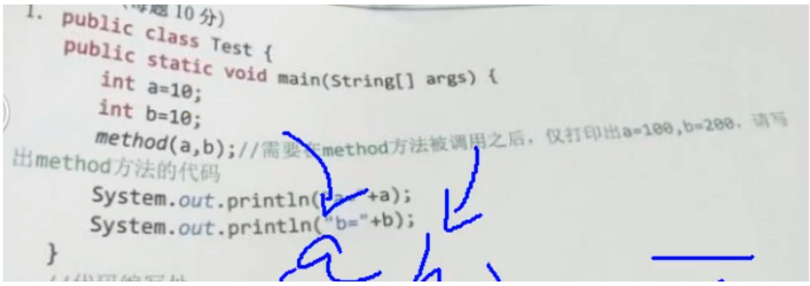


```
class Data{
    int m;
    int n;
}


main(){
    Data data = new Data();
    data.m = 10;
    data.n = 20;
     v.swap(data);
    sysout(data.m,data.n);
}
swap(Data data){
    int temp = data.m;
    data.m = data.n;
    data.n = temp;
}
```

例题2：数组的交换

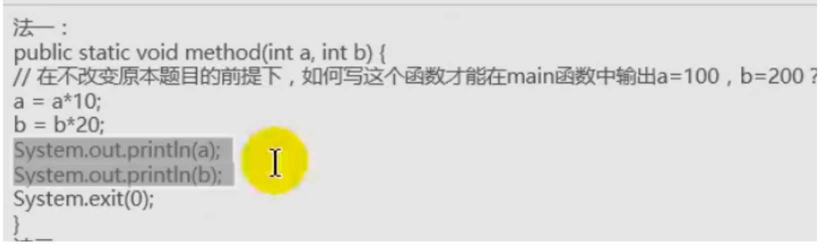
```
public class ArrayTransfer {
 public static void main(String[] args) {
 ArrayTransfer test = new ArrayTransfer();
 int[] arr = new int[]{1,6,0,5,7,2,9};
 test.printfArray(arr);
 System.out.println();
 test.sort(arr);
 test.printfArray(arr);
 }

  //遍历数组
  public void printfArray(int[] arr){
  for(int i = 0;i < arr.length; i ++){
  System.out.print(arr[i] + " ");
  }
  }

  //冒泡排序
  public void sort(int[] arr){
  for(int i = 0;i < arr.length; i ++){
  for(int j = 0; j < arr.length - 1 - i;j ++){
  if(arr[j] > arr[j + 1]){
  swap(arr,j,j+1);
  }
  }
  }
  }
//交换数组中指定两个位置的元素
public void swap(int[] arr,int i,int j){
int tmp = arr[i];
arr[i] = arr[j];
arr[j] = tmp;
}
}
```

3、经典例题



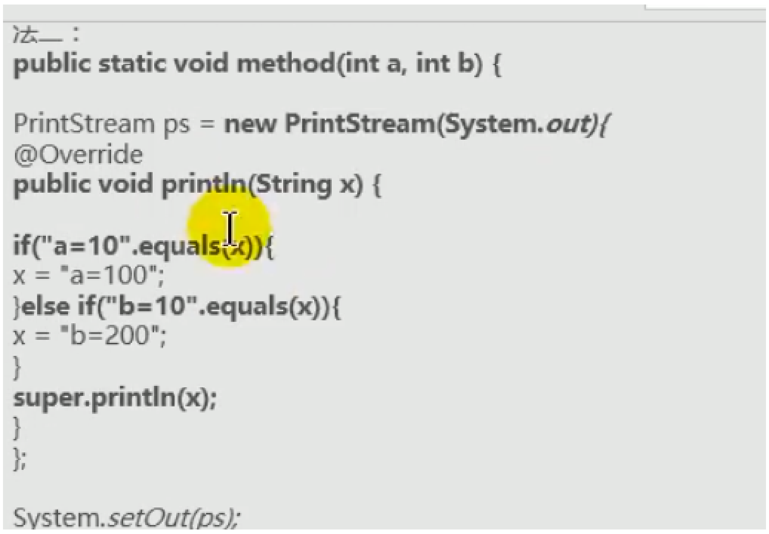
1. (每题 10 分)

```
public class Test {
public static void main(String[] args) {
    int a=10;
    int b=10;
    method(a,b);//需要在method方法被调用之后，仅打印出a=100,b=200，请写出method方法的代码
    System.out.println("a="+a);
    System.out.println("b="+b);
}
//代码编写处
```



法一：

```
public static void method(int a, int b) {
// 在不改变原本题目的前提下，如何写这个函数才能在main函数中输出a=100，b=200？
a = a*10;
b = b*20;
System.out.println(a);
System.out.println(b);
System.exit(0);
}
```



法二：

```
public static void method(int a, int b) {

PrintStream ps = new PrintStream(System.out){
@Override
public void println(String x) {

if("a=10".equals(x)){
x = "a=100";
}else if("b=10".equals(x)){
x = "b=200";
}
super.println(x);
}
};

System.setOut(ps);
```