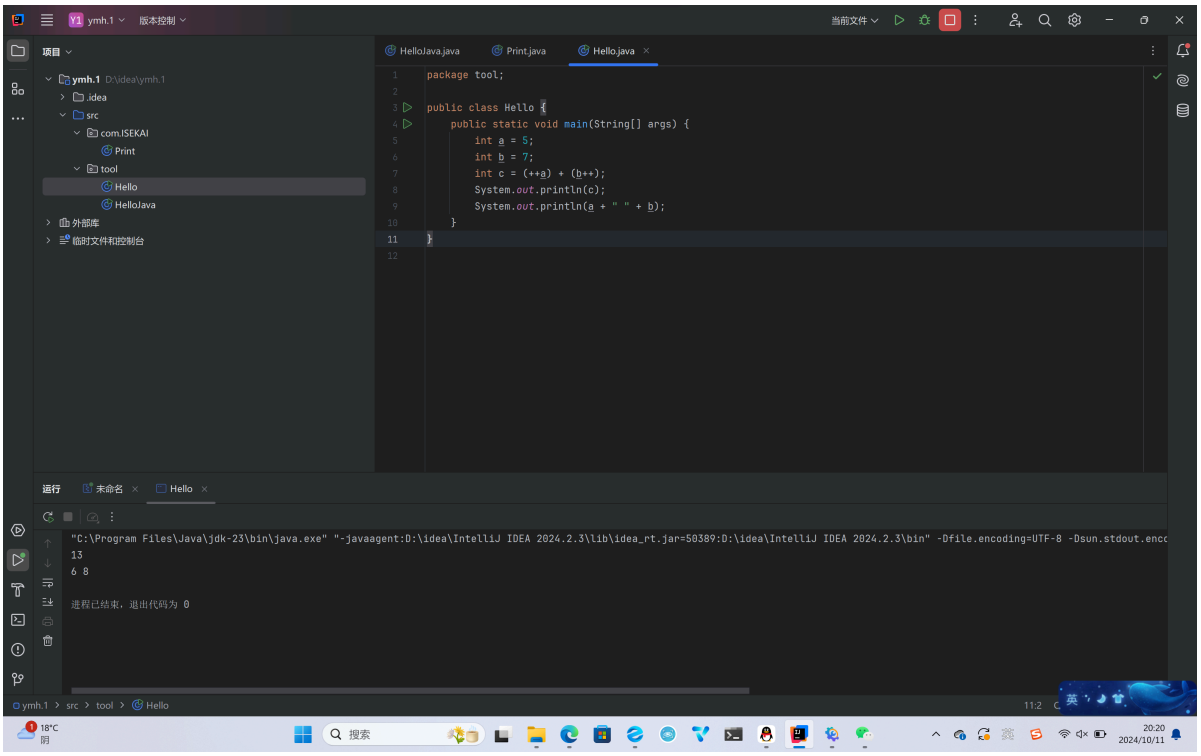
- ***基本类型***：整数类型：byte short int long　　浮点类型：float double　　字符类型：char　　布尔类型：boolean
- *byte*： 占 1 个字节，取值范围为-128 到 127。
- *short*： 占 2 个字节，取值范围为-32768 到 32767。
- *int*： 占 4 个字节，取值范围为-2147483648 到 2147483647。
- *long*： 占 8 个字节，取值范围为 -9223372036854775808 到 9223372036854775807。
- 是自动类型转换,b=52 ,字符类型 char 会自动变为 int 类型，这里字符 '0' 对应的 ASCII 值为48
- **包装类**的主要作用是将基本数据类型包装成对象，**引用类型**存储的是对象的引用地址，而不是对象本身的值，在**缓存池**中获取已经存在的对象，而不是新创建一个对象
  false
  true
  false
  原因：创建了两个不同的 Integer 对象，存储在不同的内存地址中，所以使用 == 比较会返回 false 。
  调用 valueOf 方法且值在 -128 到 127 之间时，会从整数缓存池中获取已有的 Integer 对象，所以两个变量指向同一个对象，使用 == 比较会返回 true 。
  300 不在 -128 到 127 之间，所以使用 == 比较会返回 false

  2.计算过程：

1. 首先定义了三个变量 a 、 b 和 c ，初始值分别为 5 、 7 。
2.

- ++a 是先自增操作,此时 a 的值变为 6 。
- b++ 是后自增操作，先使用 b 的当前值 7 参与运算，然后再将 b 的值加 1 。
- 所以 c = 6 + 7 = 13 。

3. System.out.println(c) 输出变量 c 的值 13
4. System.out.println(a + " " + b) 输出变量 a 和 b 的值，此时 a 的值为 6 ， b 的值为 8 ，所以输出结果为 6 8
   前自增（++i）:先将变量的值加 1，然后再使用变量的值进行其他操作。
   后自增（i++）:先使用变量的当前值进行其他操作，然后再将变量的值加 1。



3. - 正数的补码：与原码相同

-负数的补码：先将其绝对值的二进制表示取反（即每一位 0 变为 1，1 变为 0），然后再加 1

-0010

-对于任意的非负整数a， 式子a&(-a)表示的数是a的二进制表示中*最右边的1所对应的数值*。***原因：*** 当我们对a取负时, 再与a进行按位与运算，这个过程会使得除了a最右边的1及其右边的位以外，其他位都变为0