实验03

运行结果

```
Counts of blocks: 6
Counts of transactions: 11
Index:1
id of illegal transactions: a16f3ce4dd5deb92d98ef5cf8afeaf0775ebca408f708b2146c4fb42b41e1
4be
Height:4
reason is using a used input
Index:2
id of illegal transactions: f4184fc596403b9d638783cf57adfe4c75c605f6356fbc91338530e9831e9
e16
Height:4
reason is sig fail
Index:3
id of illegal transactions: 591e91f809d716912ca1d4a9295e70c3e78bab077683f79350f101da64588
073
Height:4
reason is sig fail
```

```
Index:4
id of illegal transactions: 8aa673bc752f2851fd645d6a0a92917e967083007d9c1684f9423b1005406
73f
Height:5
reanson is using an illegal output
Index:5
id of illegal transactions: f8325d8f7fa5d658ea143629288d0530d2710dc9193ddc067439de803c370
66e
Height:5
reanson is input value less than output value
(output value: 34733442>input value: 1637692)
Counts of illegal: 5
Counts of legal:6
Please enter next option(1:input height,2:input txID,3:quit):
```

分析

在原有程序的基础上，要增加对script的验证，那么首要问题就是读入script.

读入之后，应该根据具体的情况来设计栈，进行栈操作并验证.

最后综合之前的代码，得到结果。

实现

读入数据

由于上一个实验忽略了script字段的读入，所以这次应该进行修改。

注意到部分数据有script，部分数据无，所以应该增加一个判断。

通过新增一个count辅助变量来进行判断。

验证操作

要以栈的方式实现，所以要先实现一个栈。

为便于操作，以类的方式实现，由于本实验不需要用到栈的所有相关函数，所以在实现栈时主要实现了以下几个必要操作：

- `empty()` 判断是否为空。
- `top()` 返回栈顶元素。
- `pop()` 去掉栈顶元素。
- `push()` 存入栈顶元素。

使用链式存储。

然后进行验证,进行入栈操作，边读边操作。

其中由于计算表达式比较复杂，且不想转为后缀表达式，所以增加一个判断优先级的函数，直接按照中缀表达式来操作，具体实现方案：

具体操作参考了：

- 使用两个栈，一个存放操作数，另一个存放操作符
  将表达式存入两个栈中，边存边计算
- 从左往右扫描表达式，遇到操作数则直接将其放入操作数栈。
  遇到操作符时，如果优先级低于或等于栈顶操作符优先级，则从操作数栈弹出两个元素进行计算，并将计算结果存入操作数栈，继续与栈顶操作符的比较优先级。
- 如果遇到操作符高于栈顶操作符优先级，则直接入操作符栈；遇到左括号，直接入操作符栈，遇到右括号，则直接出栈并计算，直到遇到左括号。
- 循环步骤3，直到表达式全部被扫描；若扫描完成后，操作符栈不为空，则依次将操作符栈的元素出栈，同时将操作数里面的元素出栈进行计算，同样将计算结果存入操作数栈，直到操作符栈为空，最后在操作数栈的栈顶元素就是表达式的最终计算结果。
  ————————————————————
  参考原文链接：https://blog.csdn.net/qq_41032474/article/details/89715140

此外，`OP_CHECKSIG` 操作需要调用 `rsa.h` 头文件，其中的验证操作会有多余输出，所以进入其中进行一些删除操作来吧多余的输出去掉。

别的就是按照作业说明文档中的提示按部就班操作即可。

为了和之前的代码相兼容，所以做了一些调整，用返回值-4来代表未通过script验证这种情况，返回对应报错。