任务 1：简单程序

PC，IR 寄存器的作用。

PC,计数，指示指令的位置，地址。

IR，表示，储存指令。

ACC 寄存器的全称与作用。

累加器，可以存放数据。

用“LOD #3”指令的执行过程，解释 Fetch-Execute 周期。

1.pc 从 RAM 取指令 LOD #3

2.指令传入 IR，指令传入 Decoder，3 传入 MUX

3.数据 3 传入 ALU 右侧，再传入 ACC

用“ADD W” 指令的执行过程，解释 Fetch-Execute 周期。

1.PC 根据地址从 RAM 取指令 ADD W

2.指令传入 IR，再传入 Decoder

3.ALU 从 ACC 中取值 ，取入左侧

4.IR 再次访问 RAM 中的 W，从 W 中取值

5.把 W 的值读入 ALU 右侧

6.ALU 执行加法，即原本 ACC 加上 W，结果传入 ACC

“LOD #3” 与 “ADD W” 指令的执行在 Fetch-Execute 周期级别，有什么不同。

（3）点击“Binary”,观察回答下面问题

写出指令 “LOD #7” 的二进制形式，按指令结构，解释每部分的含义。

00010100 00000111 前为 LOD，导入数，后为#7，即数为 7。

解释 RAM 的地址。

指令和数据都存储在 RAM

该机器 CPU 是几位的？（按累加器的位数）

8 位

写出该程序对应的 C 语言表达。

```
int w,x,y;

w=3;

x=7;

y=w+7;
```

任务 2：简单循环

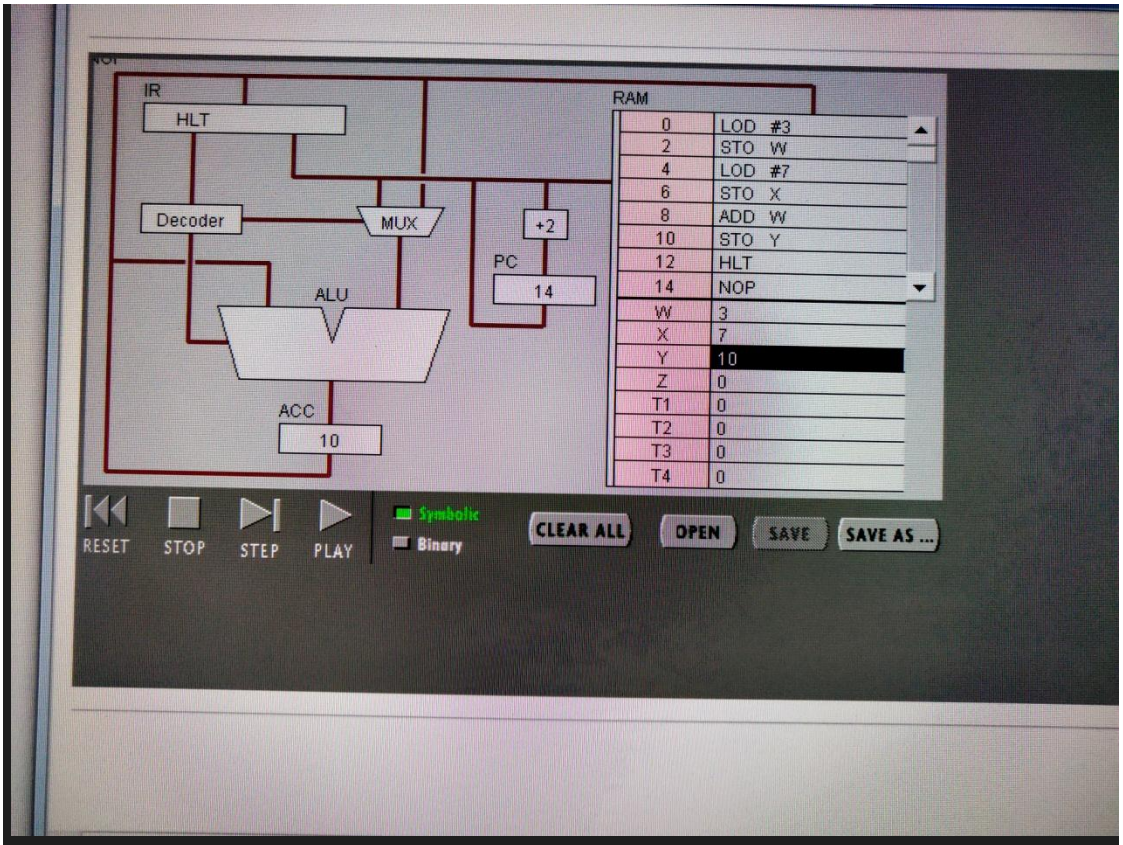
（1） 输入程序 Program 2，运行并回答问题：

用一句话总结程序的功能

给数字赋值

写出对应的 c 语言程序

```
int w,x,y;

w=3;

x=7;

y=w+7;
```



（2） 修改该程序，用机器语言实现 10+9+8+..1 ，输出结果存放于内存 Y

写出 c 语言的计算过程

```
int x=10,y;

while(x){
    y+=x;
    x--;
}
```

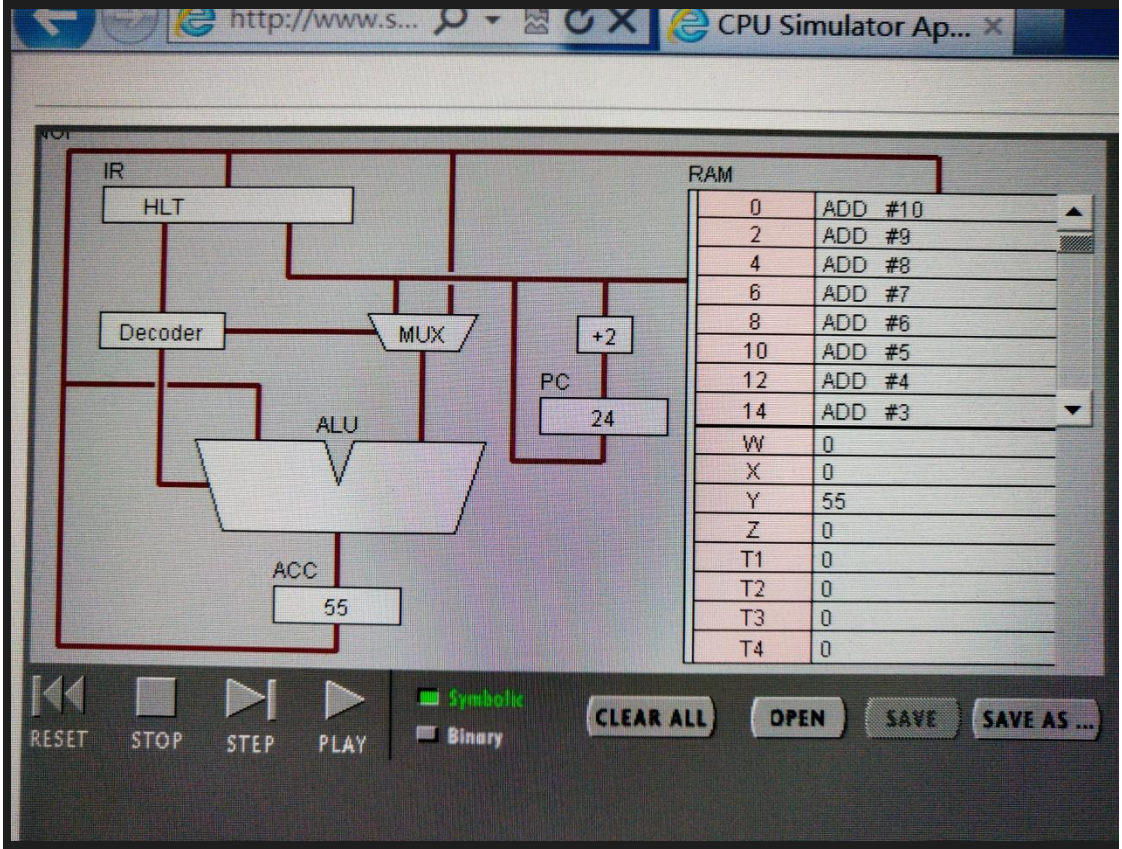
或者

```
y=10+9+8+7+6+5+4+3+2+1;
```

写出机器语言的计算过程

```
ADD #10

ADD #9

ADD #8
```

```
ADD #7

ADD #6

ADD #5

ADD #4

ADD #3

ADD #2

ADD #1

STO Y

HLT
```



用自己的语言，简单总结高级语言与机器语言的区别与联系。

联系：机器语言和高级语言都能执行相当多的简单操作，并且组合。

区别：高级语言更加直观，机器语言必须根据机器的执行顺序运行