9. 1) [illegible] $-2^{19} \sim 2^{19}-1$

2) [illegible] $-2^{11} \sim 2^{11}-1$

3) 可以，先用 lui 指令将立即数向左偏移 12 位，然后再将其存入到 rs1 中，再用 jalr 存入低12位立即数，就可实现任意 32 位指令跳转

具体的 lui, rs1, imm

jalr rd, rs1, imm2

10: 1,

①指令类型：RVC指令仅适用于特定类型的指令，例如移位指令，逻辑指令，算术指令和加载/存储指令等这些指令的共同特点是指令长度相对较小，可以用16位的指令表示。

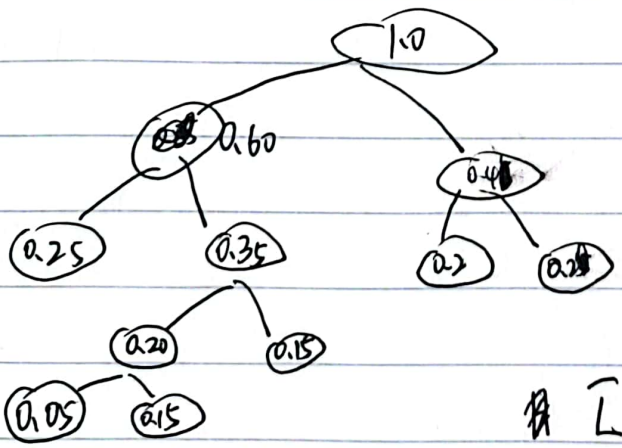
②操作数：RVC指令中的操作数数量和位数必须符合特定的规则，例如移位指令的立即数必须是5位，而算术指令中的立即数必须是6位

③操作数范围：RVC指令中的操作数范围受到限制，例如移位指令的移位量必须在0~31之间

④指令编码：RVC指令的编码格式不同于32位指令的编码格式，需要满足特定的格式要求，例如立即数需要按照特定的方式编码

2, 可以，某些操作可能需要多个寄存器来完成，例如乘除法等，RVC指令集中提供了专用的指令，例如"mul"和"div"，它们可以使用多个寄存器来完成相应的操作。

18:



$\bar{L} = 0.25 \times 2 + 0.2 \times 2 + 0.2 \times 2 + 0.15 \times 3 + 0.2 \times 4$

$= 2.55$

$\bar{r} = 1 - \frac{2.55}{3} = 0.15$

19.1, 程序为其所有储数据的栈具有一定的容量，

当函数向栈内压入的空间超过了栈的容量限制，导致向栈中写入数据时越界，此时，如果继续向栈中写数据，就会造成覆盖其他栈内的数据或者控制信息，例如函数的返回地址，导致程序计算出函数参数等，这样就会导致程序出现各种错误或崩溃。

2、1，优化代码，减少函数递归次数或减少局部变量的使用量

2，增加栈空间

3，将递归函数转化为非递归函数，这可以通过使用循环或迭代实现

4，使用堆内存代替栈内存，将需要大量空间的变量从栈内存转移到堆内存中可以避免栈溢出问题

5，使用尾递归优化，让递归函数在递归调用时不占用栈空间。

20、 Ya(F1)

...

tb(F1)

S0(F1)

Ya(F2)

tb(F2)

t1(F2)

S0(F2)

S1(F2)

...

Ya(F3)