5. 什么是 RISC-V 的 I 标准指令集中存在的 HINT 指令空间？它有什么作用？

HINT指令空间是一组用于提供给软件的提示信息的指令，可用于优化代码处理的效率，提高处理器性能

HINT指令可通知处理器执行某些操作，例如：控制分支预测、控制流水线清空、控制指令执行等

9. 回答以下问题：
   1) jal 指令包含 20 位的有符号立即数编码（J-type），该指令相较当前 PC 可以跳转的地址空间范围为多少？
   2) 条件分支指令（如 bne）包含 12 位的有符号立即数编码（B-type），这类指令相较当前 PC 可以跳转的地址空间范围为多少？
   3) 是否可以使用一条 lui 指令和一条 jalr 指令的组合完成任意 32 位绝对地址的跳转操作？

(1) $-2^{20} \sim 2^{20}-2$

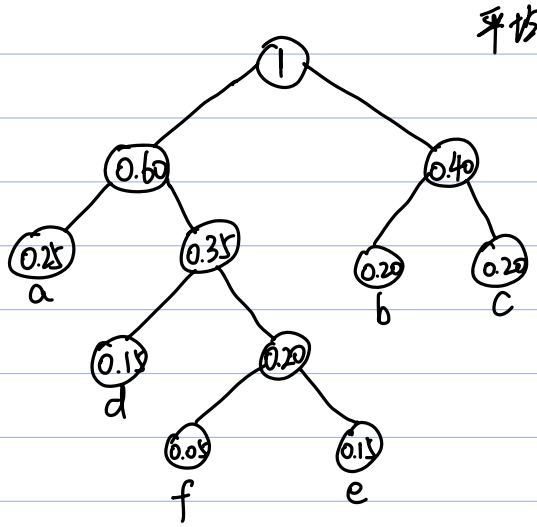
(2) $-2^{12} \sim 2^{12}-2$

(3) 可以：lui指令将目标地址的高20位写入rs1中，然后jalr指令加上低12位。

18. 有一组操作码，它们的出现几率如下表所示。

a_i	p_i
a	0.25
b	0.20
c	0.20
d	0.15
e	0.15
f	0.05

请按照霍夫曼编码对这组操作码进行编码，计算操作码的平均长度和信息冗余度。



平均长度：$\sum_{i=1}^{6} p_i l_i = (0.25+0.20+0.20)\times 2 + 0.15\times 3 + (0.05+0.15)\times 4$

$= 2.55$

冗余度：$R = 1 - \frac{-\sum_{i=1}^{6} p_i \log_2 p_i}{2.55}$

$= 0.033 = 3.3\%$

19. 回答以下问题：

1) 当函数嵌套调用层数过多（例如递归陷入死循环时），可能会造成栈溢出，请简述其原理。

2) 有什么办法可以缓解或避免特定情况下的栈溢出问题？

(1) 每一次函数的调用，都会在调用栈上维护一个独立的栈帧

每个独立的栈帧包括：函数的返回地址和参数，以及函数的局部变量

每次递归调用都会将新的栈帧压入栈帧，导致栈空间被耗尽，新的栈帧无法在栈中找到空间，就会覆盖其他数据或代码，从而导致程序异常或崩溃。

(2) 编写代码时注意减小递归调用深度、减小本地的使用量，避免出现大量连续的函数调用，或者采用非递归的方式实现嵌套调用。

此外，可以设置栈空间大小的上限来保护程序免受栈溢出的影响。

20. 假设有三个函数：F1、F2 和 F3。其中 F1 包含 1 个输入参数，计算过程使用寄存器 t0 和 s0；F2 包含 2 个输入参数，计算过程使用寄存器 t0-t1 及 s0-s1，返回一个 int 值。F1 执行过程中会调用 F2，F2 执行过程中会调用 F3。下表模拟了 F1 执行过程中栈的内容，其中第一行为 F1 函数被首次调用时 sp 寄存器指向的位置。请在表中填入当 F2 函数首次调用 F3 前栈内保存的可能内容，并在每行的括号内标注该值是被哪个函数所保存的。第一行的内容已经给出。（可根据需要增删行数）

ra (F1)
s_0 (F_1)
t_0 (F_1)
a_0 (F_1)
ra (F_2)
s_0 (F_2)
s_1 (F_2)
t_0 (F_2)
t_1 (F_2)
a_0 (F_2)
a_1 (F_2)
ra (F_3)