9. (1) jal 指令包含 20 位的有符号立即数编码 (J-type)，该指令相对于当前 PC 可以跳转的地址空间范围为多少。 next PC ← current PC + sign_extend(imm20<<1)

$2^{20} = 2^{19} \cdot 2^1 = \pm 1MB$

指令跳转范围为 ±1MB 地址空间.

(2) 条件分支指令包含 12 位有符号立即数编码，——?

$2^{12} = 4kB$

指令跳转范围为 ±4kB 地址空间

(3) 是否可以使用一条 Lui 指令和一条 jalr 指令完成任意 32 位绝对地址跳转。

Lui rd, imm20

rd ← sign_extend(imm20<<12)

jalr rd, rs1, imm12

next pc ← (rs1 + sign_extend(imm12)) & 64'hfffffffffffffffe

rd ← next pc+4.

可以使用一条 lui 指令和 jalr 指令的组合来完成任意 32 位绝对地址跳转操作.

```
Lui  t0, imm_32
ori  t0, t0, imm_32
jalr ra, t0, 0
```

10. 常用 32 位指令能被压缩到 16 位 RVC 指令条件？RVC 各类型指令是否都可以使用完整的 32 个通用整型寄存器？

RVC (RISC-V Compressed Instruction Set Extension) 是 RISC-V 指令集的一种压缩指令集扩展，通过压缩指令方式来减小代码大小，提高代码密度，从而减少指令缓存的占用和降低能耗。

需满足 ①操作数必须是 8 位有符号整数或 16 位有符号整数；

能被压缩指令

②立即数必须可以表示为 6 位或 10 位有符号整数

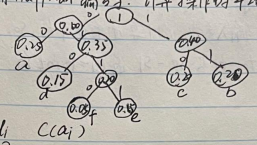
③操作数必须是通用寄存器中的一个.

RVC 可以使用完整的 32 个通用寄存器.

15. 对 整型

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

用 Huffman 编码，计算该编码的平均长度和信息冗余度.



a_i	P_i	l_i	$C(a_i)$
a	0.25	2	00
b	0.20	2	11
c	0.20	2	10
d	0.15	3	010
e	0.15	4	0111
f	0.05	4	0110

$\bar{L} = \sum_{i=1}^{6} P_i l_i = 0.25 \times 2 + (0.25 + 0.2 \times 2) \times 2 + 0.15 \times 3 + (0.15 + 0.05) \times 4 = 2.55$

$H = -\sum_{i=1}^{6} P_i \log_2 P_i = -0.25 \times \log_2 0.25 - 0.2 \times \log_2 0.2 - 0.15 \times 2 \times \log_2 0.15 - 0.05 \times \log_2 0.05 \approx 2.47$

$R = 1 - \frac{2.47}{2.55} = 0.031$

19. (1) 函数嵌套调用层数过多(如递归陷入死循环)可能造成栈溢出，简述原理.

(2) 有什么办法缓解或避免特定情况下的栈溢出问题.

(1) 栈溢出是指当一个函数调用另一个函数时，会将一些信息(例如当前函数的局部变量、参数等)存储在栈中，等到被调用函数返回后再从栈中弹出这些信息。但当函数嵌套调用层数过多时，栈的深度会超过系统所允许最大深度，导致栈溢出。这种情况，操作系统会发送一个信号给程序发生了栈溢出，并终止程序运行。

(2) ①优化算法：

②尾递归优化：对于递归函数(即最后一步是返回递归调用结果的函数)可以将递归调用转换为循环，从而避免栈溢出.

③ 将迭代代替递归：减少函数嵌套层数

④ 增大栈的大小：只能缓解不能避免.

⑤ 限制递归调用深度：设置最大深度，可能会导致运行速度变慢，或者无法处理深度超过限制的递归调用。

20. F_1、F_2、F_3. F_1包含1个输入参数，计算过程使用 t0、S0.

F_2包含2个输入参数， t0-t1 及 S0-S1 返回一个 int 值.

ra (F_1)

S0旧值 (F_1)

t0 (F_1)

F_2返回地址 (F_1)

S1旧值 (F_2)

S0旧值 (F_2)

t1旧值 (F_2)

t0旧值 (F_2)

F_3返回地址 (F_2)

S1旧值 (F_2)

S0旧值 (F_2)

t1旧值 (F_2)

t0旧值 (F_2)