1. 判断以下描述更符合 SRAM 还是 DRAM，还是都符合.

	SRAM	DRAM
(1)访问速度更快	✔	
(2)每比特需要的晶体管数目少		✔
(3)单位容量造价更便宜		✔
(4)常用作主存		✔
(5)需要定期刷新		✔
(6)断电后失去存储的信息	✔	✔
(7)支持随机访问	✔	✔

2. 已知一个双面磁盘有 2 个盘片、10000 个柱面，每条磁道有 400 个扇区，每个扇区容量为 512 字节，则它的存储容量是_**8.192**_GB

   2*2*10000*400*512 = 8,192,000,000 Byte = 8.192GB

3. 已知一个磁盘的平均寻道时间为 6ms，旋转速度为 7500RPM，那么它的平均访问时间大约为_**10**_ms

   6ms + 0.5*(60/7500*1000)ms = 10ms

4. 已知一个磁盘每条磁道平均有 400 个扇区，旋转速度为 6000RPM，那么它的平均传送时间大约为_**0.025**_ms

5. 考虑如下程序

```
for (int i = 0; i < n; i++) {
   B[i] = 0;
   for (int j = 0; j < m; j++)
      B[i] += A[i][j];
}
```

判断以下说法的正确性

(**N**) 对于数组 A 的访问体现了时间局部性。

(**Y**) 对于数组 A 的访问体现了空间局部性。

(**Y**) 对于数组 B 的访问体现了时间局部性。

(**Y**) 对于数组 B 的访问体现了空间局部性。

6. 高速缓存

   a) 一个容量为 8K 的直接映射高速缓存，每一行的容量为 32B，那么它有_**256**_组，每组有_**1**_行。

   b) 一个容量为 8K 的全相联映射高速缓存，每一行的容量为 32B，那么它有_**1**_组，每组有_**256**_行。

   c) 一个容量为 8K 的 4 路组相联映射高速缓存，每一行的容量为 32B，那么它有_**64**_组，每组有_**4**_行。

   d) 一个容量为 16K 的 4 路组相联高速缓存，每一行的容量为 64B，那么一个 16 位地址 0xCAFE 应映射在第_**43**_组内。

7. 判断以下说法的正确性

(**Y**)	保持块大小与路数不变，增大组数，命中率一定不会降低。
(**N**)	保持总容量与块大小不变，增大路数，命中率一定不会降低。
(**N**)	保持总容量与路数不变，增大块大小，命中率一定不会降低。
(**Y**)	使用随机替换代替 LRU，期望命中率可能会提高。

8. 有以下定义：

```
// 以下都是局部变量
int i, j, temp, ians;
int *p, *q, *r;
double dans;
// 以下都是全局变量
int iMat[100][100];
double dMat[100][100];
// 以下都是函数
int foo(int x);
```

如果将下列左侧代码优化为右侧代码，有没有可能有副作用？如果有请说明.

(1)	`ians = 0;`
	`for (j = 0; j < 100; j++)`
	`  for (i = 0; i < 100; i++)`
	`  ians += iMat[i][j];`
		`ians = 0;`
		`for (i = 0; i < 100; i++)`
		`  for (j = 0; j < 100; j++)`
		`  ians += iMat[i][j];`

(2)	`dans = 0;`
	`for (j = 0; j < 100; j++)`
	`  for (i = 0; i < 100; i++)`
	`  dans += dMat[i][j];`
		`dans = 0;`
		`for (i = 0; i < 100; i++)`
		`  for (j = 0; j < 100; j++)`
		`  dans += dMat[i][j];`

(3)	`for (i = 0;i < foo(100);i++)`
	`  ians += iMat[0][i];`
		`temp = foo(100);`
		`for (i = 0; i < temp; i++)`
		`  ians += iMat[0][i];`

(4)	`*p += *q;` `*p += *r;`	`temp = *q + *r;` `*p += temp;`

答:
(1) 会
(2) 不会，因为浮点数不能结合
(3) 不会，因为foo 可能有副作用
(4) 不会，如果 pqr 指向同一个元素那么两个运算不等价

9. 假设已有声明 `int i, int sum, int *p, int *q, int *r, const int n = 100, float a[n], float b[n], float c[n], int foo(int), void bar()`, 以下哪种优化编译器总是可以进行?

A	`for(i = 0; i < n; ++i){`
	`  a[i] += b[i];`
	`  a[i] += c[i];`
	`}`
		`float temp;`
		`for(i = 0; i < n; ++i){`
		`  temp = b[i] + c[i];`
		`  a[i] += temp;`
		`}`

B	`*p += *q;`
	`*p += *r;`
		`int temp;`
		`temp = *q + *r;`
		`*p += temp;`

C	`for(i = 0; i < n; ++i)` `  sum += i*4;`	`int N = n * 4;` `for(i = 0; i < N; i += 4)`

		sum += i;

D	for(i = 0; i < foo(n); ++i)
	bar();
		int temp = foo(n);
		for(i = 0; i < temp; ++i)
		bar();

C

10. 阅读下列 C 代码以及它编译生成的汇编语言

```
long func() {
    long ans = 1;
    long i;
    for (i = 0; i < 1000; i += 2)
        ans = ans ?? (A[i] ?? A[i+1]);
    return ans;
}
```

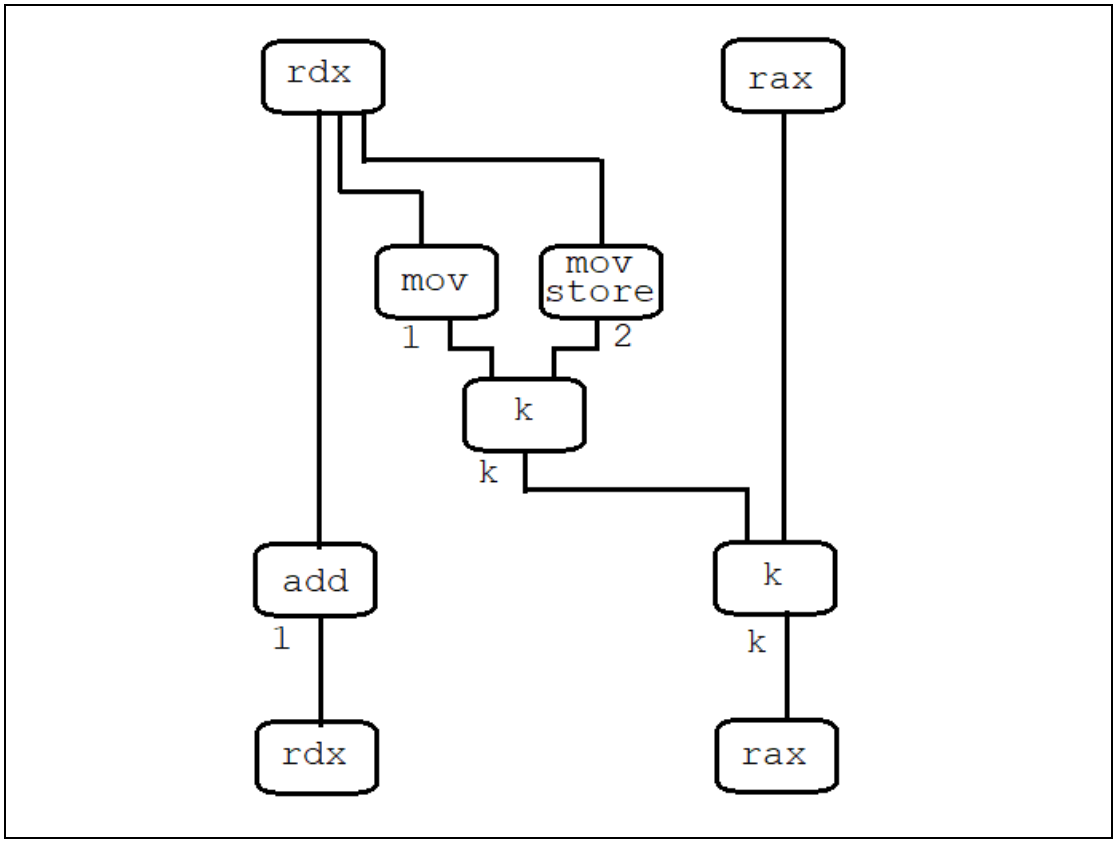
```
func:
    movl $0, %edx
    movl $1, %eax
    leaq A(%rip), %rsi
    jmp .L2
.L3:
    movq 8(%rsi,%rdx,8), %rcx // 2 cycles
    ?? (%rsi,%rdx,8), %rcx    // k + 1 cycles
    ?? %rcx, %rax             // k cycles
    addq $2, %rdx             // 1 cycles
.L2:
    cmpq $999, %rdx           // 1 cycles
    jle .L3
    rep ret
```

该程序每轮循环处理两个元素。在理想的机器上（执行单元足够多），每条指令消耗的时间周期如右边所示。

(1) 当问号处为乘法时，k = 8。此时这段程序的 CPE 为__**4**__

(2) 当问号处为加法时，k = 1。此时这段程序的 CPE 为__**0.5**__

数据相关图：


rdx
rax
mov
1
mov
store
2
k
k
add
1
k
k
rdx
rax

S=2 ⇒ index 占1位
E=2 ⇒ offset 占1位
8位地址空间 ⇒ tag 占6位

	Addr.	t	i	o	Addr.	t	i	o	
m	3	0	1	1	44	0010 11	0	0	m
m	180	1011 01	0	0					
m	43	0010 10	1	1					
h	2	0	1	0					
m	191	1011 11	1	1					
m	88	0101 10	0	0					
h	190	1011 11	1	0					
m	14	0000 11	1	0					
h	181	1011 01	0	1					

得分

176
16
192

176 + 15

16 8n +8

第五题（15 分）

Cache 为处理器提供了一个高性能的存储器层次框架。下面是一个 8 位存储器地址引用的列表（地址单位为字节，地址为 10 进制表示）：

3，180，43，2，191，88，190，14，181，44

1. (7 分)考虑如下 cache（S=2，E=2），每个 cache block 大小为 2 个字节。假设 cache 初始状态为空，替换策略为 LRU。请填补下表：
(Tag 使用二进制格式；Data 使用十进制格式，例：M[6-7]表示地址 6 和 7 对应的数据)

	V	Tag	Data	V	TAG	Data
SET 0	1		M[180-181]	1		M[44-45]
SET 1	1		M[190-191]	1		M[14-15]

共命中_____次（1 分），分别访问地址_____（地址用 10 进制表示，2 分）

解答：

答案：

	V	Tag	Data	V	TAG	Data
SET 0	1	101101	M[180-181]	1	001011	M[44-45]
SET 1	1	000011	M[14-15]	1	101111	M[190-191]

共命中 3 次

（表格上每空格 1 分，tag 和 data 都正确才得分；）

命中次数回答正确得 1 分；

分别为访问地址 2、190、181（三个地址完全正确得 2 分，答对两个得 1 分）

2. (5 分)现在有另外两种直接映射的 cache 设计方案 C1 和 C2，每种方案的 cache 总大小都为 8 个字节，C1 块大小为 2 个字节，C2 块大小为 4 个字节。假设从内存加载一次数据到 cache 的时间为 25 个周期，访问一次 C1 的时间为 3 个周期，访问一次 C2 的时间为 5 个周期。针对第一问的地址访问序列，哪一种 cache 的设计更好？（请分别给出两种 cache 访问第一问地址序列的总时间以及 miss rate）

答案：

Address	Binary address	C1 hit/miss	C2 hit/Miss
3	000000 11	M	M
180	101101 00	M	M
43	001010 11	M	M

2	000000 10	M	M
191	101111 11	M	M
88	010110 00	M	M
190	101111 10	H	H
14	000011 10	M	M
181	101101 01	H	M
44	001011 00	M	M

C1 更好。(1 分)

C1: miss rate = 8/10 = 80%, (1 分) total cycles = 8 * 25 + 10 * 3 = 230 (1 分)

C2: miss rate = 9/10 = 90%, (1 分) total cycles = 9 * 25 + 10 * 5 = 275 (1 分)

3. (3 分)现在考虑另外一个计算机系统。在该系统中，存储器地址为 32 位，并采用如下的 cache：

Cache datasize	Cache block size	Cache mode
32 KiB	8 Bytes	直接映射

此 cache 至少要占用_____Bytes. (datasize + (valid bit size + tag size) * blocks)

答案：

cache block 为 8 bytes，所以 b=3；

cache block 一共 32*1024 / 8 = 4096 个，又因为是直接映射，所以 s=12；于是 tag 位一共 t = 32 - s - t = 17。所以总大小为：

totalsize = datasize + (valid bit size + tag size) * blocks

= 32 * 1024 + (1 + 17) * 4096 / 8 = 41984 (bytes)

得分

第四题（20 分）

分析64位的Y86 ISA中新加入的条件内存传送指令：crmmovqXX和cmrmovqXX。crmmovqXX和cmrmovqXX指令在条件码满足所需要的约束时，分别执行和rmmovq以及mrmovq同样的语义。其格式如下：

rmmovq	4	0	rA	rB	D（8字节）
crmmovqXX	4	fn	rA	rB	D（8字节）
mrmovq	5	0	rA	rB	D（8字节）
cmrmovqXX	5	fn	rA	rB	D（8字节）

1．请按下表补全每个阶段的操作。需说明的信号可能会包括：icode，ifun，rA，rB，valA，valB，valC，valE，valP，Cnd；寄存器堆R[]，存储器M[]，程序计数器PC，条件码CC。其中对存储器的引用必须标明字节数。（10分）

阶段	rmmovq rA,D(rB)	cmrmovqXX D(rB),rA
取指	icode:ifun ← M_1[PC] rA:rB ← M_1[PC+1] valC ← M_8[PC+2] valP ← PC + 10	
译码	valA ← R[rA] valB ← R[rB]	
执行	valE ← valB + valC	valE ← valB + valC Cnd ← Cond(CC, ifun)
访存	M_8[valE] ← valA	valM ← M_8[valE] 或 if(Cnd) valM ← M_8[valE]
写回	none	if(Cnd) R[rA] ← valM
更新PC	PC ← valP	

2.为了执行上述新增指令，我们需要改进教材所描述的PIPE处理器，在回写（W：Write Back）阶段引入寄存器以保持流水线信号___W_Cnd___（请参考教材对信号的命名规则书写），以便有条件地更新寄存器内容。在如此改进的PIPE处理器上，请写出如下信号的HCL代码。（6分）

信号	HCL代码
F_stall	(E_icode in {IMRMOVQ, IPOPQ} || (_____①_____)) && E_dstM in _____②_____ || IRET in _____③_____
E_bubble	(_____④_____) || (E_icode in {IMRMOVQ, IPOPQ} || (_____①_____)) && E_dstM in _____②_____
M_bubble	m_stat in _____⑤_____ || W_stat in _____⑤_____

附HCL描述中的常数值编码表如下：

IHALT	halt指令的代码	INOP	nop指令的代码
IRRMOVQ	rrmovq指令的代码	IIRMOVQ	irmovq指令的代码
IRMMOVQ	rmmovq指令的代码	IMRMOVQ	mrmovq指令的代码
ICRMMOVQ	crmmovqXX指令的代码	ICMRMOVQ	cmrmovqXX指令的代码
IOPL	整数运算指令的代码	IJXX	跳转指令的代码
ICALL	call指令的代码	IRET	ret指令的代码
IPUSHQ	pushq指令的代码	IPOPQ	popq指令的代码
FNONE	默认功能码	RNONE	表示没有寄存器文件访问
ALUADD	表示加法运算	RRSP	表示%rsp寄存器ID
SAOK	正常地址操作状态码	SADR	地址异常状态码
SINS	非法指令异常状态码	SHLT	halt状态码

① E_icode == ICMRMOVQ && e_Cond

② {d_srcA, d_srcB}

③ {D_icode, E_icode, M_icode}

④ E_icode == IJXX && !e_Cond

⑤ {SADR, SINS, SHLT}

3.对于下面的Y86汇编代码，请使用上述条件内存传送指令将其修改为不带跳转的汇编代码序列。假设下面的代码片段在教材所描述的PIPE处理器上运行，不考虑该片段前后代码的影响以及高速缓存（cache）失效的情况，假设%rsi初值为0，处理器设计使用总是选择（always taken）的预测策略。原始代码片段预计运行___11__周期，改进代码片段预计执行___9(或8)___周期。（4分）

注：由于第2小题的HCL描述，要求带Cond，因为这是一个更优化的设计，而且和教材上对IJXX的考虑一致。基于此，第3小题可以有两种指令序列，cmrmoveqne在前的序列执行需要8个周期，cmrmoveqe在前的指令序列需要9个周期，因此最后一空填8或9都对。

原始代码	改进代码
andq %rsi %rsi jne L1 mrmovq 8(%rdx), %rax jmp L2 L1: mrmovq 8(%rdx), %rbx L2: addq %rax, %rbx	（9个周期的版本） andq %rsi %rsi cmrmovqe 8(%rdx), %rax cmrmovqne 8(%rdx), %rbx addq %rax, %rbx （8个周期的版本） andq %rsi %rsi cmrmovqne 8(%rdx), %rbx cmrmovqe 8(%rdx), %rax addq %rax, %rbx

得分

第五题（20 分）

现有一个能够存储 4 个 Block 的 Cache，每一个 Cache Block 的大小为 2 Byte（即 B = 2）。内存空间的大小是 32 Byte，即内存空间地址范围如下：

0_{10} (00000_2) -- 31_{10} (11111_2)

现有一程序，访问内存地址序列如下所示，单位是 Byte。

0_{10} 3_{10} 4_{10} 7_{10} 16_{10} 19_{10} 21_{10} 22_{10} 8_{10} 10_{10} 13_{10} 14_{10} 24_{10} 26_{10} 29_{10} 30_{10}

1．Cache 的结构如下图所示（S=2， E=2），初始状态为空，替换策略 LRU。请在下图空白处填入上述数据访问后 Cache 的状态。（12 分）

（TAG 使用二进制格式；Data Block 使用十进制格式，例：M[6-7]表示地址 6_{10}-7_{10} 对应的数据）

	V	TAG	Data Block	V	TAG	Data Block
set0	1	110	M[24-25]	1	111	M[28-29]
set1	1	110	M[26-27]	1	111	M[30-31]

上述数据访问一共产生了多少次 Hit： __0__ （2 分）

2．在第 1 小题的基础上，现增加一条数据预取规则：每当 cache 访问出现 miss 时，被访问地址及其后续的一个 cache block 都会被放入缓存，即当 M[0-1]访问发生 miss，则把 M[0-1]和 M[2-3]都放入缓存中。那么，这 16 次数据访问一共产生了多少次 Hit ： __8___ （2 分）

3．在第 1 小题的基础上，如果每一个 Cache Block 的大小扩大为 4 Byte（即 B = 4，cache 大小变为原来的 2 倍），这 16 次数据访问一共产生了多少次 Hit ： __8___ （2 分）

4．在第 3 小题的基础上，考虑增加 2 中的数据预取规则，这 16 次数据访问一共产生了多少次 Hit ： __12___ （2 分）