6. ①地址中间位作为组索引可以有效地将缓存划分为多个组，每个组可以存储多个数据块。这种组织方式可以更好地利用缓存空间，提高缓存的空间利用率。

②地址的高位作为标签可以更快速地进行标签比较，高位通常包含更多的位数，因此具有更大的标签空间

7 ①相同位数，可以使地址转换更简单。

②地址转换的一致性使得缓存系统对于虚拟内存系统来说更加透明和易于使用。

③. 将组索引和块内偏移与页偏移位数相匹配可提供性能优化的方法。

8.(1). $1\times 97\% + 110\times 3\% = 4.27$

(2). $\frac{64kB}{1GB} = \frac{2^6\ kB}{2^{20}\ kB} = \frac{1}{2^{14}} = 2^{-14} \approx 0$

$0\times 1 + 1\times 110 = 110$

(3). 当缓存空间足够时，局部性原理可以极大提高处理器访存性能；不够多时，空间局部性会开始延长访存性能。

(4). $x\times 1 + (1-x)\times 110 \le 105$.

$110 - 109x \le 105 \qquad x \ge \frac{5}{109} \approx 4.59\%$

平均缓存命中率大于4.59%时才会使存储系统使用L1获性能收益。

9.

32	5	21	6
8	3	23	6
1	0	26	6
256	8	18	6
64	6	19	7
256	8	18	6
64	6	20	6
32	5	20	7

10. (1). $0.22 \times (1-P_1) + 100 \times P_1 < 0.52(1-P_2) + 100 \times P_2$

$0.22 + 99.78 P_1 < 0.52 + 99.48 P_2$

$\Rightarrow \quad P_1 < 0.3\% + 99.69\% P_2 \approx 0.3\% + P_2$

(2). $0.22 \times (1-P_1) + 0.22k P_1 < 0.52(1-P_2) + 0.52k \cdot P_2$

$0.22 + 0.22(k-1) \cdot P_1 < 0.52 + 0.52(k-1) \cdot P_2$

$\Rightarrow \quad P_1 < \frac{0.3}{0.22(k-1)} + \frac{0.52}{0.22} \cdot P_2 \approx \frac{1.36}{k-1} + 2.36 P_2$

11. $0x1001 = 0001\ 0000\ 0000\ 0001$

$0x1005 = 0001\ 0000\ 0000\ 0101$

$0x1021 = 0001\ 0000\ 0010\ 0001$

$0x1045 = 0001\ 0000\ 0100\ 0101$

$0x1305 = 0001\ 0011\ 0000\ 0101$

$0x2ee5 = 0010\ 1110\ 1110\ 0101$

$0xfdf5 = 1111\ 1101\ 1111\ 0101$

直接：Index = address mod 16 = 4 6次.

2路：Index = address mod 8 = 3 3次.

4路：Index = address mod 4 = 2 3次

8路：Index = address mod 2 = 1 0次

12. 直接映射：一个block存4个数 96÷4=24(行).

缺失率：$\frac{24+16\times 99}{100\times 99}=16.24\%$.

2路组相联：25%

13.
```
for(int j=0; j<128; ++j){
    for(int i=0; i<64; ++i){
        A[j][i]=A[j][i]+1;
    }
}
```

14. (1). int数据为4B，每块存8个数，共64×128=2^{13}个数.

优化前：缺失次数为64×128=2^{13}次

优化后：$\frac{2^{13}}{8}=2^{10}$次

(2). 4KB/32B=128块.

优化前：$\frac{2^{13}}{8}=2^{10}$次

优化后：$\frac{2^{13}}{8}=2^{10}$次.

(3). 优化前：4KB×8=32KB

优化后：$2^{13}\times 4B=32KB$.