Output:

```
if no cache,time= 1100
if cache size=5,time= 780
if using LRU(cache size 5),time= 1180
if cache size=10,time= 510
if using LRU(cache size=10),time= 460
```

Discussion

A: if there is no cache, the time is 1100 time unit

B: if a small cache is used, there performance is increased by 30%

C: however, when the small cache is combined with LRU, the performance is decreased. There is cache pollution when cache size is very small, the replacement is too frequent, causing useful memory locations to be evicted out of cache

D: doubling the cache, the performance will increase by 50%

F: doubling the cache size and also using LRU, the performance will increase significantly (almost 60% as compared to no cache). The cache hit rate is increased due to larger cache size combined with LRU.

```
#name: zhengwen zhang
#SMU ID: 47502277

import random

def main():
    nocache()
    cache5()
    LRU5()
    cache10()
    LRU10()

# define the global variable memloc
memloc =   [10,11,12,13,14,15,16,80,81,82,83,84,85,86,80,81,
82,83,84,85,86,80,
            81,82,83,84,85,86,80,81,82,83,84,85,86,80,81,82,
83,84,85,86,80,81,
            82,83,84,85,86,17,18,19,20,21,22,90,91,92,93,94,
95,90,91,92,93,94,
            95,90,91,92,93,94,95,90,91,92,93,94,95,90,91,92,
93,94,95,90,91,92,
            93,94,95,90,91,92,93,94,95,23,24,27,28,29,30,31,
27,28,29,40,41,42]

def nocache():

    sum = 0

    for i in range(len(memloc)):
        sum += 10
    print 'if no cache,time=',sum

def cache5():
    cache = [0, 0, 0, 0, 0]
    total = 0

    for i in range(len(memloc)):

        if memloc[i] in cache:
            total += 1
        else:
            index = random.randint(0, 4)
            cache[index] = memloc[i]
            total += 11
    print 'if cache size=5,time=', total
```

```
def LRU5():
    #initialize the cache
    cache = [1,2,3,4,5]
    #initialize the counter of each entry
    counter = [1,3,0,4,2]
    total = 0

    for i in range(len(memloc)):
        if memloc[i] in cache:
            total += 1
            ind = cache.index(memloc[i])
            for j in range(5):
                if j == ind:
                    counter[j] = 0
                else:
                    counter[j] += 1

        else:
            total += 11
            # find the longest entry
            irep = counter.index(max(counter))
            cache[irep] = memloc[i]

            # increase all the other entries by 1
            for k in range(5):
                if k == irep:
                    counter[k] = 0
                else:
                    counter[k] += 1

    print 'if using LRU(cache size 5),time=', total

def cache10():
    cache = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
    total = 0

    for i in range(len(memloc)):

        if memloc[i] in cache:
            total += 1
        else:
            index = random.randint(0, 9)
            cache[index] = memloc[i]
            total += 11
```

```
    print 'if cache size=10,time=', total


def LRU10():
    #initialize the cache
    cache = [1,2,3,4,5,6,7,8,9,10]
    #initialize the counter of each entry
    counter = [1,3,0,4,2,5,7,6,9,8]
    total = 0

    for i in range(len(memloc)):
        if memloc[i] in cache:
            total += 1
            ind = cache.index(memloc[i])
            for j in range(10):
                if j == ind:
                    counter[j] = 0
                else:
                    counter[j] += 1

        else:
            total += 11
            # find the longest entry
            irep = counter.index(max(counter))
            cache[irep] = memloc[i]
            # increase all the other entries by 1
            for k in range(10):
                if k == irep:
                    counter[k] = 0
                else:
                    counter[k] += 1

    print 'if using LRU(cache size=10),time=', total


if __name__ == "__main__":
    main()
```