Question#1

1. 2^64 / 2^14 = 2^50
2. (2^64 / 2 ^4)\*2^14 = 2^74 bytes
3. 2^14 / 2^9 = 2^5

2^64 / 2^14 = 2^50

50 / 5 = 10

4 2^74 / 2^14 = 2 ^ 60

Question#2

1 (a): A program holding some location information can be mapped logical address to physical address in different area.

(b): MMU

(c): We want to improve the efficiency of using main memory so we should put programs in it dynamically(not continuously). If programs are relocatable, we can achieve this goal.

2 Small page size:

Advantages: saving storage and decreasing the amount of ‘holes’.

Big page size:

Advantages: page table becomes smaller then memory can store less and decrease page fault.

3 The advantage of paging over segmentation is easy to implement for fixed page size.

4 The advantages of segmentation over paging is divided program by logic by programmers, which means it can be modified easier than paging and it also can improve the utilization of main memory because of proper size allocation for each processes.

Question#3

Hashing with linear probing is better than others because we check the proper addresses can be faster.

Question#4

((1 + 0.2) \* 80% + (1 + 1 +0.2) \* (1 – 80%))\*98% + (20 \* 10^3) \* (1 - 98%) = 401.372 microseconds

Question#5

(a) 100/3/2 = 50 / 3

(b) 100 \* 100 / 3 / 2 = 5000 / 3

Question#6

LFU is Least Frequently Used algorithm and LRU is Least Recently Used algorithm. So if the processes always continuously or very high frequently ask for pages and then also asking anymore later, the LFU can generate fewer page faults than the LRU. If the processes ask for pages then a long time they don’t ask again, the LRU can generate fewer page faults than the LFU.

For example, 1 1 1 2 2 2 3 4 5 3 4, the amount we can store is 4. Because 1 and 2 is the most frequent visiting, so 3, 4, 5, 3, 4 would not replace 1 and 2 so LFU will generate a lot of page faults but LRU can fix this problem but 1 2 1 2 1 2 3 4 5 6 1 2 , in this case LFU performs better.