1. Assume the range search is based on two values, lowestkey and highestkey.
In such case, the static hash doesn’t support range search. It becomes a scan.
The B+ tree support the range search by finding the lowestkey and highestkey in the tree respectively in $O(\log_d(n))$. Then find the keys pointing to the records in the middle of lowest and highest keys by following the double links. If the file is clustered based on the key, the I/O cost is further reduced.

2. Assume the alternative 1 is used in the static hash. Each bucket contains the records not the pointers.

   a) Each entry in the primary bucket has 5 bytes that is the length of a pointer to the overflow page
   The number of blocks are $5*1000 / 1000\text{bytes} = 5$

   b) The average number of records in a bucket is $100,000/1000 = 100$
   the total bytes of records in a bucket are $100*200 = 20,000$
   Assume the number of buckets are $N$ and each bucket contains a pointer to the next block.
   $N*1000 = 20,000+5N$
   $N = 21$

   c) Based on the assumptions in b), further assume the record queried is uniformly distributed in a bucket. Then the average block accesses are $11$. If the primary directory is not in the main memory, the average block accesses are $11+1=12$

   Note:
   If you assume the alternative 2 is used in the static hash, the answer is

   1
different. Each bucket contains the (key, pointer) as a data entry. The length of the key/pointer pair is 50+5=55. The total length of data entries in a bucket is 55*100=5500. Also assume the number of blocks in a bucket is N. Then N*1000=5500+5*N. N=6.

The hash table contains pointers to the record, which is a RID. So to retrieve the record, it has to access the block containing the record. Thus the number of average block accesses is 6/3+1=3. If the primary directory is not in the main memory, the number is 3+1 = 4

3. Assume the B+ tree is at least half full and each internal node or leaf node has one bucket.

The order of the B+ tree is (blocksize-pointersize)/2(keysize+pointersize) = (4096-8)/2*(9+8) = 121

The height of the tree is \( \log_{121}(100,000,000) = 4 \).

The search needs 4 block accesses, that is 4*20 = 80ms.

4. a) a hash index on sid of student.
    b) a clustered hash index on sid of enrolled
c) a clustered hash index on sid of enrolled
d) a B+tree on (Cno, sid) of enrolled.