1.

ANS.

1)Packet over the network = 10 megabits/sec

Interrupt process time = 1 msec = 1000 micro sec

Packet size = 1024 bytes (including header)

Total time required to send a packet of 1024 bytes over the network

= OS copies data to kernel + copies data to network controller board + sent over network + after sending network controller stores each bit + CPU interruption + kernel to kernel buffer + copies to user

= 1024 micro sec +1024 micro sec + 819.2 micro sec + 819.2 micro sec + 1000 micro sec + 1024 micro sec + 1024 micro sec

= 6734.4 micro sec

= 6.73 msec Since transmission time 0.83 msec the total time of 1024 bytes is 6.93 msec

That's why per second maximum 147763 bytes can transfer.

2

ANS.

Time to read/write a memory word = 10 nsec = 10-8 sec

Number of registers = 34.

As each register is one word long, the number of words is 34.

When an interrupt occurs, all the registers are posted to the stack and after that to continue the execution all of them are popped from the stack.

This means 68 words need to read/write.

Total time = 68*10-8 sec One interrupt takes 68 * 10-8 sec

That's why this machine can handle the number of interrupts are in one sec = 1/ 68 *10-8 = 1470588.

3.

ANS.

Rotation rate = 300 RPM

Number of sectors = 8

As a disc is double interleaved it takes 2.625 (0.375 * 7) rotation to read a whole track in order from 0.

Total rotation = 0.5 + 2.625 = 3.125

Time for 3.125 rotation = 3.125/300 = 0.625 sec

Total bytes of 8 sectors = 512 * 8 = 4096 bytes.

Data rate = 4096/0.625 = 6553.6 bytes/sec

Now without interleaving it takes 0.875 (0.125*7) rotation to read whole track from 0

Total rotation = 0.5 + 0.875 = 1.375

Time for 1.375 rotation = 1.375/300 = 0.00458 min = 0.275 sec

Data rate = 4096/0.275 = 14894.54 bytes/ sec

Data rate degradation due to interleaving = 6553.6/14894.54 * 100 = 44 %

4.

ANS.

A) FCFS

Total number of cylinders travelled by arm

= (20-10)+(22-10)+(22-20)+(20-2)+(40-2)+(40-6)+(38-6)

= 146

Total seek time = 146*6

= 876 msec

B) SSTF

Total number of cylinders travelled by arm

= (20-20)+(22-20)+(22-10)+(10-6)+(6-2)+(38-2)+(40-38)

= 60

Total seek time = 60*6

= 360 msec

C) SCAN

Total number of cylinders travelled by arm

= (20-10)+(22-20)+(38-22)+(40-38)+(40-10)+(10-6)+(6-2)

=58

Total seek time = 58*6

= 348 msec

5.

ANS.

The filename can be appended with its directory's name. For example: /harsh/520/text.txt

6.

ANS.

Each block size is 1024 bytes means 10 block size is 10240 bytes.

Now if the ith node contains indirect address then number of blocks

= Each block size/number of bytes

= 1024/4

= 256

Maximum file size is 256KB(direct address + indirect address)

256 block size = 256 *1024 = 262144 bytes.

The possibility of the size of the largest file is 262144 + 10240 = 272384 bytes.

7.

ANS.

After file A uses six blocks bitmaps: 1111 1110 0000 0000

a) File B uses five blocks.

Bitmap : 1111 1111 1111 0000

b) A is deleted.

Bitmap : 1000 0001 1111 0000

c) C uses eight blocks.

Bitmap : 1111 1111 1111 1100

d) B is deleted.

Bitmap : 1111 1110 0000 1100

8.

ANS.

In computing, a cache is a high-speed data storage layer which stores a subset of data, typically transient in nature, so that future requests for that data are served up faster than is possible by accessing the data's primary storage location. Caching allows you to efficiently reuse previously retrieved or computed data.

It is more effective to cache files using a virtual storage system rather than a physical storage cache. This is due to the fact that virtual memory is a data caching solution that is both efficient and lightweight. The operating system can then detect page faults and determine whether or not the page is resident (in memory). The operating system loads from the hard drive if it is not already in memory. Because all pages load at the same moment, this can happen quite quickly. If a page is resident, the operating system stores it in memory until it is needed by the program. When the program needs the page, the operating system intercepts the request and loads it from memory.

As a result, the virtual storage system is able to handle all page faults at all times. With physical memory caches, this isn't always the case. All page faults should be caught and verified to see if the operating system's data is still present in memory. Otherwise, the data from the floppy disk must be loaded into memory. Even if the data is in memory, the operating system should always complete this action. Because the file is constantly present (in memory), the operating system

can cache it. The operating system is not required to check for the existence of the file. You must catch the page fault and load the file from the disk if the operating system does not cache the file in memory. If you cache the file in physical memory, you'll need to catch any page faults, make sure the file is resident, and make sure the file isn't corrupted.

9.

ANS.

The goal was to improve performance by caching disk blocks in memory, therefore avoiding having to go to the disk while reading or writing data. Before the advent of unified buffer caching, a cached buffer was identified by a device number and a block number. Modern operating systems, including Mac OS X, use a unified approach wherein in-memory contents of files reside in the same namespace as regular memory.

The UBC conceptually exists in the BSD portion of the kernel. Each vnode corresponding to a regular file contains a reference to a ubc_info structure, which acts as a bridge between vnodes and the corresponding VM objects. Note that UBC information is not valid for system vnodes (marked as VSYSTEM), even if the vnode is otherwise regular. When a vnode is createdsay, because of an open() system calla ubc_info structure is allocated and initialized.

10.

ANS.

Files stored on a disk by their inode(index node), not by their name. An inode contains information about the actual data and the location on the hard disk. But inode doesn’t contain any information regarding the file name. Therefore, directories help to map the file names to their inode numbers.