B

Department of Computer Science and Engineering
Final Examination Spring 2025
CSE 321: Operating Systems

Duration: 2 Hours **Total Marks:** 45

Answer the following questions.
Figures in the right margin indicate marks.

Name:	**ID:**	**Sec:**

---

1. **Write the answers in the question paper.**

CO2

a) **Identify** the error in the code below: **[1]**

```
1 do
2    {
3        wait(chopstick[i]);
4        wait(chopstick[(i++) % 5]);
5
6        /* eat for sometime */
7
8        signal(chopstick[i]);
9        signal(chopstick[(i+1)%5]);
10    } while (true);
```

Answer:

b) **Identify** which method below will get stuck first. **[1]**

```
sem1 = 10;
sem2 = 0;
mutex=1;
```

```
method1(){
      while(true){
            wait(sem1);
            wait(mutex);
            /*Do some
operations*/
            signal(mutex);
            signal(sem2);
      }
}
```

```
method2(){
      while(true){
            wait(mutex);
            wait(sem2);
            /*Do some
operations*/
            signal(mutex);
            signal(sem1);
      }
}
```

Answer:

2. **[1]**
CO3

a) **Write the answer in the question paper.**
A CPU is translating logical addresses to physical addresses for every memory access. To avoid the high latency of looking up the page table each time, it caches recent translations in a small, fast lookup table. **Mention** what hardware tool is the CPU using to speed up logical-to-physical address translation by caching recent page table entries?

Answer:

b) **Write the answer in the answer script.** **[3]**
Consider a system with a TLB miss ratio of **4.5%**, and the TLB access time of **23 ns**. If Effective Access Time is **293 ns**, then **calculate** the main memory access time.

c) **Write the answer in the answer script.**

In a system the size of the main memory is **64 Bytes** and size of each page is **8 Bytes**. In a certain moment the CPU generates logical addresses **13, 26, 4** and **22** of a process respectively. Logical memory and page table of the process is given below.

Logical Memory	
0	chmod
1	touch
2	rm
3	mv

PMT	
0	5
1	0
2	7
3	15

Main Memory

i. **Calculate** corresponding physical addresses of generated logical addresses. If the physical address is not valid then write "**invalid**". **[4]**

ii. **Map** user's view of the main memory. **[2]**

d) **Write the answer in the answer script.** **[10]**

Shoumo has implemented an OS which has a page size of **4 KB** and each entry in the page table is sized **1 Byte**. He has installed a new game and the new process has the logical address space of **64 bits**. Now in order to fit the pages of the process in the main memory, his OS will apply Two-Level Paging technique in outer page number bits until the outermost page table can be allocated in a frame of the main memory.

Illustrate the logical address space of the process including the necessary outer page bits, inner page bits and offset bits of every step with proper mathematical calculations during the paging mechanism of the system described above.

e) **Write the answer in the answer script.** **[6+1 =7]**

Dipu was designing a system in which there are **4 frames** in the main memory. In a particular scenario main memory needs to accommodate **12 pages** according to the order of the given reference string.

[7, 3, 2, 1, 2, 3, 7, 8, 3, 5, 8, 2]

Now as Dipu needs to implement his system very fast, he mistakenly introduced a bug in the implementation of his LRU algorithm. According to his algorithm:

- When a page needs to be replaced, that page and the page stored in the next frame to it get replaced together.
- If that page is the only page in the list, only that page is removed.

Apply this buggy Optimal algorithm in order to accommodate the pages in the main memory and **find** out the hit ratio.

3. CO3

a) **Write the answers in the question paper.** **[3]**

i) Kabbya was working on a file named log.txt. To perform a write operation on the file, the system must complete three steps:

1) Write the updated content to a data block
2) Update the file's inode to reflect the new size and block pointers
3) Update the corresponding bit in the data bitmap

While performing this write, the system unexpectedly lost power and crashed. After rebooting, Farid checked the file and noticed that the file size had increased, but the file system checker informed that there was an inconsistency. Based on this post-crash state, **deduce** which of the following writes were successful and which ones failed:

- Data block write
- Data bitmap update
- Inode update.

Answer:

Data block write	
Data bitmap update	
Inode update	

b) **Write the answer in the answer script.** **[3]**

A file system uses UNIX inode data structure which contains **12** direct block addresses, **3** single indirect blocks and **2** triple indirect blocks. The size of each block is **64 Bytes** and the size of each block address is **8 Bytes**. **Find** the number of double indirect blocks if the maximum possible file size is **82.25 KB**.

c) **Write the answer in the answer script.** **[4]**

A file system has inode size = **256 Bytes** and block size = **2 KB**. First **5** blocks contain superblock, journal, group descriptor, data bitmap and inode bitmap. Now **calculate** the address of the inode number **28**. The disk is sector addressable and the size of each sector is **256 Bytes**. Find out the sector address of the inode.

d) **Write the answer in the answer script.** **[4]**

An existing file named "**h**" is allocated in **2** data blocks. In this file new data will be written after the existing contents and after the operation **2** more data blocks will be allocated to the file.

- Path of the file: "**/nf/h**"
- To write in the file it was opened first by **open()** system call.
- After opening the file, **write()** system call was issued in the file to write new contents in the file and after the write operation **2** new data blocks were allocated for the file.

Illustrate the file access path timeline including proper order of all the operations according to the scenario described above.

4. **Write the answer in the answer script.** **[2]**

CO1 According to copyright, **draw** the updated access matrix after one possible change we can make in (D2, F1) index.

Object / Domain	F1	F2	F3
D1	Owner	Read	Read*
D2	Read*	Write*	Owner
D3	Write		Write