1) Total marks – 25; 2) Unless stated otherwise, all addresses are assumed to be in decimal system (for ease of calculations)

Q1. Give an example of time sharing/multiplexing and space sharing/multiplexing in OS **[0.5*2 = 1]**

Q2. Mention 3 goals of the OS for memory management. **[1.5]**

Q3. Assume correct headers are present. Assume the VA of i is 0x100100. Print the output of the following program with explanation. **[1]**

```
int i = 2; int ret = fork();
if (ret != 0) {
        wait(NULL); printf("%p\n",(int*)&i); i = 4;printf("%d\n", i); sleep(10);}
else    { printf("%p\n", (int*)&i); i = 3; printf("%d\n", i);}
```

Q4. When are the base & bounds registers setup? What happens to them during a context switch? Why is base and bounds approach called dynamic relocation **[0.5+0.5+1]**

Q5. Given a Base register value of 1000 and Bound register value of 300, find the physical address corresponding to the following virtual addresses: a) 1000, b) 1200, c) 200 **[0.5*3]**

Q6. Give pros and cons of base & bounds, segmentation, paging, paging + TLB, multi-level paging. **[5]**

Q7. Assume a 16 KB VA space. We have 5 segments: S1, S2, S3, S4 and S5. 16 KB VA can be represented via 14 bits. Each of these segments is 2KB in size. Divide the VA into the 5 segments (+ free space), such that the top "x" most significant bytes can help determine the segment and the remaining bits tell the offset. Many such mappings would be possible. Give one such mapping. **[1]**

Q8. How can we use segmentation to share memory across processes to save memory? We can reduce fragmentation to null if we allocate each byte of VA to a byte of PA. Why is this a bad idea? **[0.5+0.5]**

Q9. Assume a 64 bytes address space. This can be represented using 6 bits. What is the smallest and the largest page size that we can have for this address space? What are the corresponding number of pages? How does internal fragmentation and memory overhead (page table size) of the page table change with increase in page size? **[2]**

Q10. When does a linear page table waste memory? How does a linked list reduce memory overheads? At what expense does this reduction in memory usage in the Linked list page table come at? **[1]**

Q11. Assume that VA can be expressed in 32 bits. Each page is 1024 bytes. Each page table entry takes 8 bytes. Assuming a linear page table – how many bits are used for VPN and Offset? How many PTEs can we store in a single page? Now, if you want to use a multi-level page table, with the following goal: to make each piece of the page table fit within a single page. How many levels would you need in the page table and how you divide the 32 bits across these levels? **[5]**

Q12. For the workload: [0, 1, 2, 0, 1, 3, 0, 3, 1, 2, 1], find the hit rate for LRU, Oracle and most recently used (MRU) for cache size = 3 **[3]**