Mananshi Vyas
AU1940289

OS End-Sem Practical Exam

Description and Logic of Code

Q1 b)

Thread is a single sequence stream within a process. Threads share their code, data and OS resources but have their own counter, register and stack. Multithreading can help us use parallelism which allows the computer to run multiple programs concurrently.

For this question, we would create three threads, one for lowercase alphabets, the second for uppercase alphabets and the last for digits. We would insert all these values in a shared vector and then call the vector in a particular order to create the pattern given below.

A 1 a B 2 b C 3 c ... Y 25 y Z 26 z

Q2 b)

The buddy system is a memory allocation/deallocation and management algorithm that manages memory in the power of two increments. If the memory size is 2^m and size M is required,

- If 2^m-1 < M <= 2^m, it allocates the whole block
- Else it recursively divides the block into equal parts and if the condition is satisfied it allocates the block.

For example, the system has a space of 128KB. For the 18KB process, it will first divide into two equal parts of 64, then one of which would get divided into 32KB blocks each. If the blocks are divided further it won't be able to fit the 18KB process, hence 32KB block is allotted.

Allocation of memory can be done using free lists and we can use an unordered set for deallocation. The starting address segment stores the key and the size of the segment stores the value. This gets updated every time a request for an allocation is made. In case a deallocation request is made, the set would be checked for validity and then the block would be added to the free list. Also, if its buddy is free, the blocks would be merged and will be placed on the free list above them.

Q3

In Producer-Consumer Problem, producer and consumer share affixed size buffer which is used as a queue. The producer is required to generate data and put that in the buffer while the

consumer is required to consume the data from the buffer. We have to make sure that the producer doesn't try to put data into the buffer while it's full and that the consumer doesn't try to consume the data while the buffer is empty. We would initialize a mutex, 2 semaphores for empty and full slots and a buffer array. The producer code would initially wait as there are no empty slots and then we would insert an item in the buffer array and then we would wake the consumer and inform that some data has been added which needs to be consumed. The consumer code would initially wait as there would be no free slots. It will then remove an item from the buffer array and then a signal would be passed to the producer informing that the buffer slots have been emptied.