Technical University of Denmark

Written Examination, December 11, 2025

Course: Operating Systems

Course No.: 02159

Aids allowed: All aids

Exam duration: 4 hours

Weighting: According to their respective points (100 points in total)

Notes:

- All questions contribute toward the final grade. The maximum number of points awarded by each correctly answered question is listed next to the question.
- For short questions, the following apply: (i) a brief answer is expected; (ii) you are also expected to briefly explain your answer in approximately 2-3 lines; (iii) a correct answer with a correct explanation yields 4 points in total.
- For long questions, the following apply: (i) there might be more than one correct answer; (ii) you are expected to briefly explain your answer in approximately half a page; (iii) a complete and correct answer yields 12 points; (iv) if you make additional assumptions, remember to briefly explain them.

A. Short Questions

Question A.1: What is the key advantage of microkernel operating systems compared to monolithic operating systems? Explain briefly your answer. (**4 points**)

Question A.2: You are developing an application that is composed of multiple threads. Each thread updates a global variable. Which method you would use to avoid a race condition? Explain briefly your answer. (**4 points**)

Question A.3: Provide a real-world example of an application where the NFU (Not Frequently Used) page replacement algorithm will perform very poorly. Do not use the examples provided in the textbook or slides. Explain briefly your answer. (**4 points**)

Question A.4: A program is executed and the current state of the memory is as shown in Figure 1. The next two instructions add two integers: the first integer is in memory address 50000 and the second integer is in memory 50004. How many page faults are raised? Explain briefly your answer. For simplicity, you can assume that the addresses shown in the figure are in thousands: for example 4K-8K is equal to 4000-8000. (**4 points**)

Question A.5: What will the code in Figure 2 print? Explain briefly your answer. You can assume that all system call invocations are successful. (**4 points**)

Question A.6: What will the code in Figure 3 print? If the code is executed multiple times, will the printed numbers always be in the same order? Explain briefly your answer. You can assume that all system call invocations are successful. (**4 points**)

Question A.7: What will the code in Figure 4 most likely print? Explain briefly your answer. You can assume that all system call invocations are successful. (**4 points**)

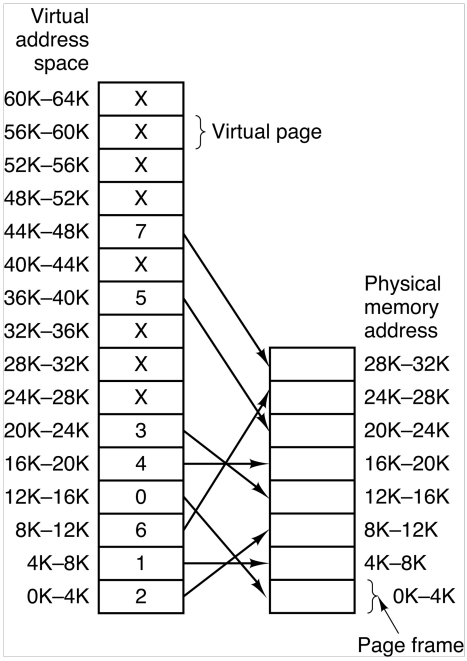

Figure 1: Data for Question A.4.

```
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>

int counter = 1;

void child(void) {
    counter++;
    exit(0);
}

int main(void)
{
  if (fork() == 0) {
    child();
  }
  waitpid(-1, NULL, 0);

  counter++;
  printf("%d\n", counter);
  return 0;
}
```

Figure 2: Code of Question A.5.

```c
#include <stdio.h>
#include <pthread.h>

pthread_t thread_id[5];

void* count(void *a) {
  printf("%d\n", *(int*)a);
  pthread_exit(NULL);
}

int main(void) {
  int i;
  for (i=1;i<=5;i++) {
    pthread_create(&thread_id[i], NULL, count, &i);
    pthread_join(thread_id[i], NULL);
  }
  return 0;
}
```

Figure 3: Code of Question A.6.

```c
#include <stdio.h>
#include <pthread.h>

int A = 0;
pthread_t thread_id[1000];

void* count(void *input) {
  int i;
  for (i=0;i<1000;i++)
    A++;
  pthread_exit(NULL);
}

int main(void) {
  int i;
  for (i=0;i<1000;i++)
    pthread_create(&thread_id[i], NULL, count, NULL);
  for (i=0;i<1000;i++)
    pthread_join(thread_id[i], NULL);
  printf("%d\n", A);
  return 0;
}
```

Figure 4: Code of Question A.7.

```c
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>

int main(void){
  fork();
  fork();
  printf("a\n");
  exit(0);
}
```

Figure 5: Code of Question A.8.

Question A.8: How many times will the letter 'a' be printed if we execute the code in Figure 5? Explain briefly your answer. You can assume that all system call invocations are successful. (**4 points**)

Question A.9: A typical example of a deadlock appears when making transactions on multiple accounts protected by multiple mutexes, i.e., each account is protected by a mutex. One solution that is safe from race conditions and deadlocks is having one single mutex over all the accounts. What is the disadvantage of this solution? Explain briefly your answer. (**4 points**)

Question A.10: Is Direct Memory Access (DMA) better to be used when the operating system is busy serving a large number of I/O-bound processes or a large number of CPU-bound processes? Explain briefly your answer. (**4 points**)

Question A.11: Describe an OS functionality that is important for user experience to be implemented using interrupt-based I/O. Justify briefly your answer. (**4 points**)

Question A.12: A program needs to find something in a very large array of data. The parent process divides the search space in four pieces and spawns four child processes. Each child process searches one piece of the search space in parallel. The child process that finds it reports back to the parent process. Everything mentioned up to here is already implemented. Your task is to implement the final part that follows: The parent process needs to stop the other three child process that are still searching. Which interprocess communication method you would use to implement this functionality? Which system call corresponds to it? Briefly explain your answer.

Question A.13: In which circumstances mutual exclusion with a spinlock is good for efficiency? Explain briefly your answer. (**4 points**)

Question A.14: What is the main advantage of preemptive scheduling compared to non-preemptive scheduling? Explain briefly your answer. (**4 points**)

Question A.15: What is the main advantage of multiprocessor systems compared to single-processor systems? Explain briefly your answer. (**4 points**)

Question A.16: Why communication protocols are necessary for implementing distributed applications over the Internet. Explain briefly your answer. (**4 points**)

B. Long Questions

Question B.1 (**12 points**)

Let's assume that you are designing a server for the OS Challenge. Everything is as specified on the OS Challenge specification document with one important difference: the incoming requests have variable difficulty (i.e., the difference between the `start` and the `end` is not constant). How would you design the scheduler? Motivate your answer.

Question B.2 (12 points)

(i) Summarise the advantages and disadvantages of processes and threads.

(ii) You are developing a Domain Name System (DNS) server. The purpose of a DNS server is to translate domain names (e.g. `www.dtu.dk`) to IP addresses (e.g. `192.38.84.35`). When a DNS server receives a request from a client, it first looks if this domain name is in a local cache, otherwise it makes a request to a higher-tier DNS server. Once it retrieves the IP address, it responds to the client. The DNS server should handle multiple requests from potentially hundreds of clients in parallel. Would you implement the DNS server using multiple processes (e.g. one process per client) or multiple threads (e.g. one thread per client)? Motivate your answer and discuss if the disadvantages of your solution are relevant in this use case. If relevant, also discuss how you would overcome them.

Question B.3 (12 points)

After the presentation of the experiments of all other groups, propose a new experiment for the OS Challenge that your group has not tested before. Motivate the experiment by explaining why you believe it will improve the performance and describe how you would test if your hypothesis is true.