КИЇВСЬКИЙ СТОЛИЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ БОРИСА ГРІНЧЕНКА
Факультет інформаційних технологій та математики
Кафедра інформаційної та кібернетичної безпеки
імені професора Володимира Бурячка

Затверджено на засіданні кафедри
інформаційної та кібернетичної безпеки
імені професора Володимира Бурячка
(протокол №13 від 16.10.2026)

РОБОЧА ПРОГРАМА ІСПИТУ

СИСТЕМНЕ ПРОГРАМУВАННЯ

галузь знань	12 Інформаційні технології
спеціальність	123 Комп’ютерна інженерія
освітня програма	123.00.01 Комп’ютерна інженерія

2026-2027 навчальний рік

Опис програми іспиту

Київський столичний університет імені Бориса Грінченка
Кафедра інформаційної та кібернетичної безпеки імені професора Володимира Бурячка
Програма іспиту з дисципліни «Системне програмування»
3 курс – освітній рівень – перший (бакалаврський)
Спеціальність 123 Комп'ютерна інженерія
Освітня програма: 123.00.01 Комп'ютерна інженерія
Форма проведення: тестування на платформі Moodle в ЕНК дисципліни: https://elearning.kubg.edu.ua/course/view.php?id=26843
Тривалість проведення **1 год. 10 хв.**
Максимальна кількість балів: **40 балів**
Екзамен проводиться в університетській аудиторії у тестовій формі із використанням персональних комп'ютерів, якщо ситуація дозволяє проведення освітнього процесу офлайн. Якщо ж освітній процес проходить дистанційно, то екзамен проводиться онлайн в режимі відеоконференції засобами Google Meet. Студент дає відповіді на запитання електронного тесту в системі Moodle. Іспит складається з тестової та прикладної частини. Тест містить 30 тестових питань закритого типу (вибір правильної відповіді із запропонованих варіантів), які передбачають автоматичну (комп'ютерну) перевірку і оцінюються по 1 бали кожне. Прикладна частина містить 2 завдання відкритого типу, які передбачають перевірку та оцінку екзаменатором і оцінюються по 5 балів кожне. Екзамен проводиться із суворим дотриманням принципів академічної доброчесності, що передбачає недопустимість списування, фальсифікацій та обману. При порушенні студент відсторонюється від подальшого проходження екзаменаційного тесту із підсумковою оцінкою Fx за дисципліну. Підсумкова оцінка в балах (максимально 100 балів) за дисципліну є сумою результату поточного контролю за семестр (60 балів) та відповіді на екзамені (40 балів).
Перелік тем, які виносяться на іспит: 1. Системне програмування: визначення, значення, роль у контексті загального програмування. 2. Взаємодія апаратного та програмного забезпечення; системні інтерфейси, апаратно-залежний код, низькорівневі операції. 3. Відмінності між системним та прикладним програмуванням; рівні абстракції, доступ до апаратних ресурсів, цілі та методології розробки. 4. Компоненти системного програмного забезпечення: операційні системи, компілятори, асемблери, лінкери, завантажувачі. 5. Мови програмування низького рівня: C, C++, Assembly; їх особливості та застосування у системному програмуванні. 6. Ядро та користувацький простір: механізми захисту, рівні привілеїв, режим ядра та режим користувача.

7. Системні виклики; механізми реалізації системних викликів, ABI (Application Binary Interface).
8. Взаємодія між додатками та ядром ОС: контекстні перемикання, пастки (traps), переривання (interrupts).
9. Управління пам'яттю; фізична та віртуальна пам'ять, адресний простір процесу, механізми трансляції адрес.
10. Віртуальна пам'ять; сторінки, таблиці сторінок, буфер швидкого транслятора адрес (TLB).
11. Пейджинг та сегментація; методи управління пам'яттю, сегментні регістри, захист пам'яті.
12. Алгоритми заміщення сторінок: FIFO, LRU, NRU, алгоритм другої нагоди (Second Chance).
13. Витоки пам'яті та переповнення буферів: виявлення, запобігання, інструменти аналізу, безпекові ризики.
14. Процеси та потоки; концепції, моделі виконання, блок керування процесом (PCB), контекст процесу.
15. Планування процесів; алгоритми планування CPU, пріоритети, тайм-кванти.
16. Проблеми конкурентності; взаємне блокування (deadlocks), лайвлоки, стан гонки (race conditions).
17. Міжпроцесова комунікація (IPC); пайпи, сокети, черги повідомлень, спільна пам'ять.
18. Сокети та мережеве програмування; моделі взаємодії, стек протоколів TCP/IP, протоколи UDP та TCP.
19. Файлові системи; структури даних файлових систем, індексні вузли (іноди), методи організації файлів.
20. Буферизований та небуферизований ввід/вивід; системні виклики, стандартні бібліотеки, механізми буферизації.
21. Асинхронний ввід/вивід; неблокуючі операції, мультиплексування вводу/виводу, epoll, select, poll.
22. Драйвери пристроївl моделі драйверів, керування апаратними пристроями, взаємодія з апаратурою.
23. Управління ресурсамиl дескриптори файлів, обробка помилок, звільнення ресурсів.
24. Налагодження системних програм; використання gdb, strace, ltrace, техніки відладки.
25. Профілювання та оптимізація продуктивності; інструменти профілювання, аналіз продуктивності, оптимізація коду.
26. Статичний та динамічний аналіз, інструменти як valgrind, address sanitizer.
27. Захист пам'яті, механізми SELinux, AppArmor, контроль доступу (ACL).
28. Запобігання переповненням буферів, ін'єкціям коду, валідація введення, використання канарок.
29. Нові мови у системному програмуванні: Rust, переваги у безпеці пам'яті, відсутність витоків.

30. Майбутні тенденції у системному програмуванні: паралельні та розподілені системи, низькорівневе програмування для IoT.

Приклад екзаменаційного завдання

1. Який механізм використовується для зменшення кількості контекстних перемикань у багатопотоковому середовищі під час синхронізації потоків?
   a) М'ютекс (Mutex)
   b) Семафор (Semaphore)
   c) Спінлок (Spinlock)
   d) Бар'єр (Barrier)
   e) Монітор (Monitor)

2. Що таке TLB (Translation Lookaside Buffer) у контексті управління пам'яттю?
   a) Кеш-пам'ять для збереження часто використовуваних даних
   b) Таблиця сторінок процесу в оперативній пам'яті
   c) Буфер для швидкого переводу віртуальних адрес у фізичні
   d) Механізм для управління витоками пам'яті
   e) Алгоритм для оптимізації пейджингу

3. Реалізуйте на мові C програму, яка створює дві нитi (потоки) за допомогою POSIX Threads (pthread). Перша нитка повинна записувати послідовність чисел Фібоначчі у спільний буфер, а друга нитка повинна читати ці числа з буфера та виводити їх на екран. Забезпечте синхронізацію між нитями, щоб уникнути умов гонки, використовуючи м'ютекси та умовні змінні.
   Вимоги**:**
   a. Використайте pthread_mutex_t для взаємного виключення доступу до спільного буфера.
   b. Використайте pthread_cond_t для координації запису та читання між нитями.
   c. Забезпечте коректне створення та завершення нитей, а також ініціалізацію та знищення синхронізаційних примітивів.
   d. Програма повинна коректно працювати без станів гонки та взаємоблокувань (deadlocks).

Екзаменатор Богдан БЕБЕШКО

Завідувач кафедри Павло СКЛАДАННИЙ