Міністерство освіти і науки України
Національний університет водного господарства та природокористування

Навчально-науковий інститут кібернетики, інформаційних технологій та інженерії
Кафедра обчислювальної техніки

04-04-300М

МЕТОДИЧНІ ВКАЗІВКИ

до лабораторних робіт з навчальної дисципліни
«Крос-платформне програмування»
для здобувачів вищої освіти
першого (бакалаврського) рівня
за освітньо-професійною програмою
«Комп'ютерна інженерія»
спеціальності 123 «Комп'ютерна інженерія»
та освітньо-професійною програмою
«Інформаційна безпека»
спеціальності 125 «Кібербезпека та захист інформації»
денної та заочної форми навчання

Рекомендовано
науково-методичною радою
з якості ННІКІТІ
Протокол № 4 від 24.02.2025 р.

Рівне – 2025

Методичні вказівки до лабораторних робіт з навчальної дисципліни «Крос-платформне програмування» для здобувачів вищої освіти першого (бакалаврського) рівня за освітньо-професійною програмою «Комп'ютерна інженерія» спеціальності 123 «Комп'ютерна інженерія» та освітньо-професійною програмою «Інформаційна безпека» спеціальності 125 «Кібербезпека та захист інформації» денної та заочної форми навчання. [Електронне видання] / Бойчура М. В. – Рівне : НУВГП, 2025. – 43 с.

Укладач: Бойчура М. В., к.т.н., доцент кафедри обчислювальної техніки.

Відповідальний за випуск: Сидор А. І., к.т.н., в.о. завідувача кафедри обчислювальної техніки.

Керівник групи забезпечення
спеціальності
123 «Комп'ютерна інженерія» Сидор А. І.

Керівник групи забезпечення
спеціальності
125 «Кібербезпека та захист інформації» Назарук В. Д.

ЗМІСТ

Вступ

Дисципліна «Крос-платформне програмування» присвячена вивченню принципів розробки програмного забезпечення, здатного працювати на різних платформах. Основна увага приділяється використанню сучасних технологій та інструментів, таких як Unreal Engine 4 (UE4), C++, Blueprints, а також методологій організації командної роботи, що включають використання GitHub і Jira.

Дана дисципліна формує навички створення інтерактивних додатків, що охоплюють ігрову індустрію і не тільки. В процесі навчання студенти освоюють архітектуру крос-платформних систем та організацію збереження й передачі даних. Завдяки практичній спрямованості, студенти отримують можливість працювати з реальними задачами, що відповідають актуальним вимогам ринку праці.

Ключові аспекти дисципліни:

- розробка ігрових світів з використанням UE4;
- основи програмування на C++ у контексті крос-платформних додатків;
- використання Blueprints для швидкої розробки;
- інтеграція командних інструментів (GitHub, Jira) у навчальний процес;
- застосування принципів об’єктно-орієнтованого програмування.

Результатом навчання є здатність розробляти інтерактивні додатки, управляти життєвим циклом проєкту та ефективно працювати в команді.

Лабораторна робота №1
Розробка найпростішої гри
(4 бали)

Мета

Розробити найпростішу гру.

Завдання

1. *(90% балів)* Побудувати ігровий світ, який міститиме наступне: 2 приміщення, стелю, освітлення, сходи, траву, провалля та воду. Для прикладу див. рис. 1.
2. *(10% балів)* Організувати можливість доступу персонажа до всіх локацій ігрового світу.

Рис. 1. Приклад ігрового світу

Хід виконання

1. Створення нового проєкту:
   - відкрийте Epic Games Launcher та встановіть Unreal Engine 4 (UE4);
   - оберіть стандартний пресет “First Person”.
2. Створення ігрового світу:

- використовуйте “Landscape Tool” для створення трави та ландшафту;
- додайте приміщення, стелю, сходи, воду та провалля за допомогою “Brush Editing”.

3. Освітлення:
   - додайте “Directional Light” для природного освітлення;
   - налаштуйте “Sky Sphere” для покращення атмосфери.
4. Рух персонажа:
   - додайте “Player Start”;
   - перевірте, що персонаж може вільно переміщуватися по світу.
5. Збереження рівня:
   - натисніть “Save All”, щоб зберегти всі зміни.

Додаткова інформація

Процес виконання даної лабораторної роботи є порівняно простим та інтуїтивно зрозумілим. Проте окремі речі є не завжди очевидними. У таких випадках доцільно користуватись мовними моделями штучного інтелекту. Для спрощення доцільно використовувати віртуальний асистент “Unreal Engine 4 Expert” чату ChatGPT. При цьому, варто деталізовано та недвозначно формулювати запити, щоб штучний інтелект правильно їх трактував.

Оскільки студенти працюватимуть із асистентом, який спеціально навчений на тестових даних по Unreal Engine 4, то назву ігрового двигуна у запиті можна не згадувати. Тоді як важливо уточнити, що Вас цікавить відповідь лише на основі Blueprints. Далі наведено приклад майже підходящого запиту.

Працюю лише з блупринтами. Я повний новачок. Користуюсь шаблоном гри "Від першого лиця". В шаблоні

вже є одне приміщення. Допоможи покроково виконати 2 завдання:

1.(90% балів) Побудувати ігровий світ, який міститиме наступне: 2 приміщення, стелю, освітлення, сходи, траву, провалля та воду.

2.(10% балів) Організувати можливість доступу персонажа до всіх локацій ігрового світу.

Недолік такого запиту полягає у тому, що віртуальному асистенту не повідомлено із якою версією Unreal Engine 4 працюємо, через що у відповіді можуть бути пропозиції виконувати інструкції, які виявляються некоректними. Наприклад, при роботі з версією 4.27.2 може виникнути така фраза-відповідь «*У вікні Modes (Швидке додавання → Basic) знайди Box Brush*», яка насправді є некоректною. Щоб уникнути таких неприємних випадків, варто у текст запиту добавити й версією Unreal Engine 4 (4.27.2). Ще може бути важливим повідомити чату, що «*У вікні Modes відсутній елемент Box Brush*». Можливі й інші нюанси, які варто уточнювати при спілкуванні з чатом з метою його налаштування для зручної роботи із цією та наступними лабораторними роботами.

Контрольні запитання

1. Яка основна функція Epic Games Launcher?
2. Як встановити UE4 через Epic Games Launcher?
3. Що таке стандартний пресет в UE4?
4. Які типи пресетів доступні в UE4?
5. Як створити новий проєкт в UE4?
6. Що таке «Blueprint» у UE4?
7. Які налаштування проєкту необхідно виконати після його створення?
8. Що таке ігровий світ в UE4?
9. Як вибрати тип проєкту при його створенні?

10. Які вимоги до системи для роботи з UE4?
11. Як відкрити ігрову сцену в UE4?
12. Що таке «Starter Content»?
13. Як працювати зі стандартними світловими джерелами в UE4?
14. Які компоненти можуть бути додані до стандартної сцени?
15. Як додати камеру до сцени?
16. Що таке «Content Browser» в UE4?
17. Чим відрізняється пресет «Blank» від інших?
18. Яким чином можна створити і налаштувати світ на основі пресету «First Person»?
19. Що таке «World Outliner» і як ним користуватися?
20. Як зберегти зміни в ігровому світі?
21. Що таке «Default Pawn Class»?
22. Як змінити основні налаштування гри в проєкті?
23. Що таке «Game Mode»?
24. Як додати об’єкт на сцену?
25. Що таке «Viewport»?
26. Як працювати з «Player Start»?
27. Як включити візуалізацію статистики продуктивності?
28. Як зібрати проєкт для тестування?
29. Як видалити непотрібні об’єкти зі сцени?
30. Що таке «Level Editor»?

Лабораторна робота №2
Знайомство з Blueprints
(4 бали)

Мета

Навчитись працювати з тригерами.

Завдання

1. *(20% балів)* Здійснювати телепорт персонажа щоразу, як тільки він впаде на певну глибину.
2. *(80% балів)* Використовуючи тригери розробити смугу перешкод, в якій куби падають зверху вниз та заважають переміститись з точки А в точку В.

Хід виконання

1. Телепорт персонажа:
   - створіть новий “Blueprint Class” типу “Actor”;
   - додайте “Trigger Box” та налаштуйте подію “OnActorBeginOverlap” для телепорту персонажа на задані координати.
2. Смуга перешкод:
   - використовуйте “Timeline” для створення анімацій руху кубів зверху вниз;
   - застосуйте “Branch” для умовного виконання логіки.
3. Тестування:
   - переконайтеся, що телепорт працює після падіння персонажа;
   - перевірте, чи куби блокують шлях від точки А до точки В.

Додаткова інформація

Досить деталізовані та точні інструкції зачасту дають популярні моделі штучного інтелекту у випадку «традиційного» програмування. Тоді як при візуальному програмуванні значно важче отримати коректну відповідь. Причина у тому, що штучному інтелекту доведеться генерувати картинку або описувати послідовність кроків замість пропозиції до бездумного копіювання результату запиту. При цьому, варіант із формуванням зображення у якості відповіді зазвичай призводить до некоректних результатів (ілюстративних зображень). Тому доводиться при виконанні лабораторних робіт очікувати від мовних моделей штучного інтелекту відповідей лише у вигляді інструкцій.

Контрольні запитання

1. Що таке Blueprints в UE4?
2. Як створити новий Blueprint Class?
3. Як створити подію у Blueprint?
4. Що таке «Event Graph» у Blueprint?
5. Як додати тригер для телепорту персонажа?
6. Що таке “Trigger Box”?
7. Як можна дізнатися координати об’єкта у просторі?
8. Як працювати зі змінними у Blueprint?
9. Що таке «Timeline» і як його використовувати?
10. Як додати смугу перешкод на основі Blueprint?
11. Як змусити об’єкти рухатись зверху вниз?
12. Як створити логіку взаємодії з об’єктами?
13. Що таке «Event BeginPlay»?
14. Як працювати з функціями у Blueprint?
15. Як відобразити текстове повідомлення на екрані?
16. Як використовувати логіку умов у Blueprint?
17. Що таке «Branch» і як його налаштувати?

18. Як створити цикл у Blueprint?
19. Як додати до гри логіку завершення рівня?
20. Як додати анімацію руху об’єктів?
21. Що таке «For Loop» у Blueprint?
22. Як перевірити колізію об’єктів через Blueprint?
23. Як налаштувати таймер у Blueprint?
24. Як реалізувати обробку події знищення персонажа?
25. Як додати зміну освітлення через Blueprint?
26. Що таке «Custom Event» у Blueprint?
27. Як додати ефекти до об’єктів через Blueprint?
28. Що таке “Spawn Actor” і як його використовувати?
29. Як тестують логіку у Blueprint?
30. Як відкрити інший рівень через Blueprint?

Лабораторна робота №3
Робота в команді
(4 бали)

Мета

Навчитись працювати в команді.

Завдання

1. *(30% балів)* Створити акаунт на GitHub, встановити GitHub Desktop та налаштувати GitHub під командну розробку.
2. *(30% балів)* Створити акаунт на Jira та налаштувати його під командну розробку.
3. *(40% балів)* Вимістити на Jira 10 завдань та назначити їм StoryPoints.

Хід виконання

1. Налаштування GitHub:
    - зареєструйте акаунт на GitHub;
    - завантажте та встановіть GitHub Desktop;
    - створіть репозиторій, додайте перший commit.
2. Налаштування Jira:
    - зареєструйте акаунт на Jira;
    - створіть проєкт та налаштуйте Kanban-дошку;
    - додайте 10 завдань із призначеними Story Points.
3. Синхронізація:
    - перевірте, що Jira інтегрована з GitHub;
    - додайте спільну роботу до репозиторію.

Контрольні запитання

1. Як створити акаунт на GitHub?
2. Як встановити GitHub Desktop?
3. Що таке репозиторій?

4. Як додати репозиторій у GitHub Desktop?
5. Як синхронізувати репозиторій з віддаленим сервером?
6. Що таке «Commit»?
7. Як зробити перший Commit?
8. Як створити акаунт на Jira?
9. Що таке «Backlog» у Jira?
10. Як створити нове завдання у Jira?
11. Що таке Story Points у Jira?
12. Як призначити завдання учасникам команди?
13. Що таке «Issue» у GitHub?
14. Як створити нову гілку у GitHub?
15. Як працювати з Pull Requests?
16. Що таке «Merge» і як його виконати?
17. Як реалізувати співпрацю учасників в репозиторії?
18. Що таке «Kanban-дошка»?
19. Як працювати з Kanban-дошкою у Jira?
20. Що таке «Sprint» у Jira?
21. Як оцінити завдання у Story Points?
22. Як перевірити статус виконання завдань у Jira?
23. Як встановити інтеграцію між GitHub та Jira?
24. Як вирішити конфлікт злиття у GitHub?
25. Як додати опис до завдання у Jira?
26. Що таке «Collaboration» у GitHub?
27. Яким чином використовують Git при роботі в команді?
28. Як додати README до репозиторію?
29. Як працювати з Issues у GitHub?
30. Як аналізувати хід виконання проєкту засобами Jira?

Лабораторна робота №4
Інтеграція персонажа в гру
(4 бали)

Мета

1. Навчитись шукати 3D-персонажів.
2. Навчитись інтегровувати персонажів в UE4.

Завдання

1. *(10% балів)* Знайти 3D модель гуманоїда, наприклад, на Sketchfab.
2. *(20% балів)* Замінити стандартного персонажа на знайденого.
3. *(70% балів)* Засобами Retarget Manager інтегрувати анімацію.

Хід виконання

1. Завантаження моделі:
   - знайдіть гуманоїда на Sketchfab або Mixamo;
   - завантажте модель у форматі .FBX.
2. Імпорт моделі:
   - використовуйте “Content Browser” для імпорту в проєкт;
   - переконайтеся, що скелет моделі сумісний із UE4.
3. Анімація:
   - відкрийте “Retarget Manager” та налаштуйте скелет;
   - інтегруйте анімацію через “Animation Blueprint”.
4. Тестування:
   - перевірте коректність рухів персонажа у грі.

Контрольні запитання

1. Де можна знайти безкоштовні 3D-моделі персонажів для гри?
2. Що таке Sketchfab і як ним користуватися?
3. Як імпортувати 3D-модель персонажа в UE4?
4. Що таке Retarget Manager в UE4?
5. Як змінити стандартного персонажа на завантажену ззовні модель?
6. Що таке «Skeleton» у контексті UE4?
7. Як налаштувати скелет персонажа для використання анімації?
8. Що таке «Animation Blueprint»?
9. Як додати анімацію до нещодавно завантаженого персонажа?
10. Як перевірити коректність анімації?
11. Що таке «Rigging» і як його використовувати?
12. Як створити власний «Pose Asset»?
13. Як налаштувати анімацію бігу персонажа?
14. Яким чином реалізовується перехід між станами анімації?
15. Що таке «State Machine» в Animation Blueprint?
16. Як налаштовується поведінка персонажа за допомогою Blueprints?
17. Як додати анімацію взаємодії персонажа з об’єктом?
18. Що таке «Retargeting Options» і як їх застосовувати?
19. Як забезпечити плавність переходів між анімаціями?
20. Як реалізувати анімацію стрибка?
21. Як додати і налаштувати звуки руху персонажа?
22. Що таке «Blend Space» і для чого він використовується?
23. Як налаштувати синхронізацію рухів персонажа з анімаціями?
24. Як створити анімаційний Blueprint для гуманоїда?

25. Що таке «Montage» у контексті анімацій?
26. Як працює Play Animation у Blueprints?
27. Як зберегти налаштування анімації для повторного використання?
28. Як користуватись Retarget Manager’ом?
29. Як перевірити сумісність моделі та анімації?
30. Що таке «Additive Animation» і як її використовувати?

Лабораторна робота №5
Віджети
(4 бали)

Мета

1. Навчитись працювати з віджетами.
2. Навчитись працювати з подіями.

Завдання

1. *(90% балів)* Додати віджети HP та MP.
2. *(10% балів)* Налаштувати отримання ушкоджень.

Хід виконання

1. Створення віджетів:
    - відкрийте “Widget Blueprint”;
    - додайте “Progress Bar” для HP і MP.
2. Зв’язування даних:
    - використовуйте “Bindings” для оновлення стану HP та MP в реальному часі;
    - додайте функцію, яка зменшує HP після отримання шкоди.
3. Тестування:
    - переконайтеся, що шкала HP/MP оновлюється під час гри.

Контрольні запитання

1. Що таке віджети в UE4?
2. Як створити новий віджет в UE4?
3. Що таке «Widget Blueprint»?
4. Як додати текст до віджета?
5. Що таке «Canvas Panel» і як її використовувати?
6. Як розташувати елементи у віджеті?
7. Як змінити стиль тексту у віджеті?

8. Що таке «Progress Bar» і як його налаштувати?
9. Як створити віджет для відображення HP гравця?
10. Як додати до віджета відображення MP?
11. Що таке «Binding» у віджетах?
12. Як зв’язати параметри персонажа з відображенням у віджеті?
13. За яким принципом організовується оновлення інформації у віджеті в реальному часі?
14. Що таке «Event Construct» у Widget Blueprint?
15. Як додати функцію для віджета через Blueprint?
16. Що таке «Visibility» у віджетах?
17. Як змінити видимість віджета?
18. Як додати кнопку до віджета?
19. Як обробити натискання кнопки через Blueprint?
20. Що таке «Z-Order» і як він впливає на віджети?
21. Як налаштувати розміри елементів віджета?
22. Як створити меню гри через віджети?
23. Що таке «HUD» і як він працює в UE4?
24. Як додати віджет до HUD?
25. Як налаштувати відображення ушкоджень у віджеті?
26. Як протестувати роботу віджетів?
27. Що таке «Anchors» у віджетах?
28. Як інтегрувати звуки у віджети?
29. Як працювати зі шрифтами у віджетах?
30. Як створити віджет, який динамічно змінює свій вигляд?

Лабораторна робота №6
Додавання бота у гру
(4 бали)

Мета

1. Навчитись програмувати переміщення бота.
2. Навчитись програмувати взаємодію гравця та бота.

Завдання

1. *(10% балів)* Знайти в інтернеті та добавити на карту ще одного гуманоїда, щодо якого зробити ретаргетинг.
2. *(10% балів)* Організувати слідування інтегрованого персонажа за гравцем.
3. *(40% балів)* Організувати нанесення шкоди при наближенні бота до гравця.
4. *(40% балів)* Організувати нанесення шкоди гравцем боту.

Хід виконання

1. Створення бота:
    - додайте нового персонажа через “AI Controller”;
    - використовуйте “NavMesh Bounds Volume” для пересування.
2. Програмування логіки:
    - реалізуйте слідування за гравцем через “Behavior Tree”;
    - налаштуйте атаку бота через “Damage Event”.
3. Тестування:
    - перевірте, чи бот наносить шкоду гравцю при наближенні.

Контрольні запитання

1. Як додати нового персонажа (бота) до проєкту?

2. Що таке «AI Controller» у UE4?
3. Як налаштувати бота для слідування у певну точку?
4. Як використовувати «NavMesh Bounds Volume» для навігації?
5. Як створити маршрут для бота?
6. Що таке «Behavior Tree» і як він працює?
7. Як додати бота на карту?
8. Що таке «Blackboard» у контексті AI?
9. Як змусити бота слідувати за гравцем?
10. Що таке «Perception System» в UE4?
11. Як для бота налаштувати критерій успіху при русі до цілі?
12. Як заборонити боту виходити за межі певної області?
13. Як забезпечити телепорт бота?
14. Що таке «Simple Move to Actor»?
15. Як створити анімації для бота?
16. Як додати можливість отримання ушкодження від бота при його наближенні до гравця?
17. Що таке «Damage Event»?
18. Як організувати взаємодію між ботом і персонажем?
19. Як перевірити відстань між ботом і гравцем?
20. Як створити функцію нанесення ушкоджень ботом?
21. Як налаштувати колізії для бота?
22. Як змусити бота змінити маршрут при появі перешкоді?
23. Що таке «AI Tasks»?
24. Як відобразити стан бота у віджеті?
25. Як зробити бота агресивним?
26. Як налаштувати швидкість руху бота?
27. Що таке «AI Debugging Tools»?
28. Як працює «OnSeePawn» у Perception System?
29. Як створити логіку втечі бота від гравця?
30. Як діяльність бота на карті?

Лабораторна робота №7
Основи програмування на C++ в UE4
(4 бали)

Мета

Ознайомитися з інтеграцією та використанням C++ у середовищі UE4.

Завдання

1. *(10% балів)* Налаштувати проєкт для роботи з C++ в UE4.
2. *(20% балів)* Створити клас C++ для управління простим об’єктом (наприклад, кубом).
3. *(40% балів)* Засобами C++ реалізувати лічильник очок, які повинні збільшуватись під час взаємодії з об’єктом.
4. *(30% балів)* Реалізувати виведення очок на екран з використанням функцій по взаємодії з віджетами в C++.

Хід виконання

1. Налаштування C++:
   - додайте новий C++ клас типу “Actor”;
   - використовуйте “UStaticMeshComponent” для візуалізації об’єкта.
2. Лічильник очок:
   - створіть функцію для збільшення очок при взаємодії;
   - виведіть результат через віджет.
3. Тестування:
   - переконайтеся, що очки правильно підраховуються.

Контрольні запитання

1. Як налаштувати проєкт для роботи з C++ в UE4?

2. Які кроки потрібно виконати для створення класу C++ в UE4?
3. Як створити базовий клас, що успадковує Actor?
4. Як додати компонент Static Mesh до класу C++?
5. Яким чином можна змінити колір об'єкта через C++?
6. Що таке UStaticMeshComponent і для чого він використовується?
7. Як засобами C++ підключити виведення тексту у віджети?
8. Як працювати з функцією Tick() у C++ класах UE4?
9. Що таке "Blueprintable Class" в UE4?
10. Як забезпечити зв'язок між Blueprints та C++ класами?
11. Як обробляти події натискання клавіш у C++?
12. Які C++ методи можна використовувати для взаємодії з об'єктами в UE4?
13. Як налаштувати точку входу для взаємодії в C++?
14. Що таке PrimaryActorTick?
15. Як підключити модульний матеріал до статичного об'єкта засобами C++?
16. Що таке "Component Hierarchy" в UE4?
17. Яким чином реалізувати множинну взаємодію з об'єктом засобами C++?
18. Як можна налагодити код C++ у UE4?
19. Що таке "Reflection System" в UE4 і як вона використовується в C++?
20. Які переваги використання C++ у порівнянні з Blueprints в UE4?
21. Як створити подію зміни кольору у C++?
22. Які модифікатори доступу використовуються в C++ класах?
23. Що таке UObject і як він використовується?
24. Як створити новий клас через Unreal Editor?

25. Що таке “Garbage Collector” в UE4?

26. Чим відрізняється UProperty від звичайних змінних у C++?

27. Що таке GetWorld() і як його використовувати в C++?

28. Як підключити додаткові модулі до проєкту C++?

29. Як створити і зареєструвати подію у C++?

30. Як створити користувацький лог для налагодження?

Лабораторна робота №8
Створення інтерактивного ігрового середовища з використанням C++
(4 бали)

Мета

Навчитися програмувати об’єкти та їхню поведінку в UE4 за допомогою C++.

Завдання

1. *(20% балів)* Реалізувати об’єкт, який з використанням C++ змінює свій розмір під час взаємодії з гравцем.
2. *(30% балів)* Реалізувати тригер, який засобами C++ запускає анімацію або змінює колір об’єкта при наближенні гравця.
3. *(50% балів)* З використанням C++ створити генератор різного виду перешкод, які з’являються випадковим чином.

Хід виконання

1. Зміна розміру об’єкта:
   - додайте клас C++ типу “Actor”;
   - реалізуйте функцію, яка змінює розмір об’єкта при натисканні клавіші або взаємодії.
2. Анімація та колір:
   - використовуйте “Dynamic Material Instance” для зміни кольору об’єкта;
   - створіть тригер, який запускає анімацію або змінює колір при наближенні гравця.
3. Генератор перешкод:
   - створіть функцію, яка випадковим чином додає перешкоди на карту;
   - використовуйте “Spawn Actor” для появи перешкод.

4. Тестування:
    - Перевірте всі сценарії взаємодії та переконайтеся, що вони працюють коректно.

Контрольні запитання

1. Як запрограмувати фізичну взаємодію об’єктів через C++?
2. Як змінити матеріал об’єкта в реальному часі через C++?
3. Як реалізувати систему взаємодії через C++ та Blueprints?
4. Як працювати з компонентами “UStaticMeshComponent” у C++?
5. Як додати фізику до об’єкта через C++?
6. Як створити подію натискання клавіші в C++?
7. Як додати систему збереження для параметрів об’єкта?
8. Як реалізувати зміни розміру об’єкта при натисканні клавіші?
9. Як працювати з “FVector” в C++ у контексті руху об’єктів?
10. Як створити динамічні текстури у C++?
11. Як засобами C++ змусити об’єкт змінювати свою орієнтацію в просторі?
12. Як реалізувати зміну кольору об’єкта при натисканні клавіші?
13. Як працювати з “UMaterialInstanceDynamic” у C++?
14. Як реалізувати підбір предмета через C++?
15. Як засобами C++ налаштувати взаємодію з декількома об’єктами одночасно?
16. Як створити анімовану зміну об’єкта засобами C++?
17. Як засобами C++ реалізувати зміну швидкості руху персонажа при взаємодії з об’єктом?

18. Як передати подію від одного об’єкта до іншого через C++?
19. Як засобами C++ реалізувати тригер взаємодії між двома об’єктами?
20. Як забезпечити відображення параметрів об’єкта в UI через C++?
21. Як засобами C++ зберігати стан об’єкта між рівнями гри?
22. Як засобами C++ використовувати “Tick()” для оновлення параметрів об’єкта?
23. Як засобами C++ організувати рух об’єкта по заданій траєкторії?
24. Як працювати з “BlueprintCallable” у C++?
25. Як реалізувати систему підбору предметів засобами C++?
26. Як взаємодіяти з анімацією об’єкта через C++?
27. Як засобами C++ налаштувати звук при взаємодії з об’єктом?
28. Яким чином передаються параметри між різними класами об’єктів у C++?
29. Як працювати з “GEngine->AddOnScreenDebugMessage” для виведення debug-інформації?
30. Як засобами C++ створити кастомний компонент для інтерактивного об’єкта?

Лабораторна робота №9
Взаємодія між персонажами за допомогою C++
(4 бали)

Мета

Дослідити реалізацію взаємодії між NPC та гравцем засобами C++.

Завдання

1. *(10% балів)* Засобами C++ створити NPC з базовою анімацією.
2. *(40% балів)* Виключно засобами C++ реалізувати поведінку NPC: переслідування або уникнення гравця.
3. *(50% балів)* Засобами C++ реалізувати систему діалогів із NPC, де гравець може вибирати репліки через віджет.

Хід виконання

1. Створення NPC:
   - додайте новий клас персонажа у C++;
   - створіть базову анімацію для NPC (наприклад, ходьбу).
2. Поведінка NPC:
   - реалізуйте логіку переслідування або уникнення гравця через “AI Controller”;
   - використовуйте “Behavior Tree” для налаштування поведінки.
3. Система діалогів:
   - додайте віджет для вибору реплік у діалогах;
   - зв’яжіть логіку діалогу з подіями через C++.
4. Тестування:
   - перевірте коректність поведінки NPC та діалогів.

Контрольні запитання

1. Як створити новий клас для персонажа у C++?
2. Як реалізувати штучний інтелект NPC на C++?
3. Яким чином передаються дані між персонажами через C++?
4. Що таке AI Controller і як його використовувати без Blueprints?
5. Як реалізувати поведінкові дерева (Behavior Tree) засобами C++?
6. Як реалізувати систему діалогів через C++?
7. Як взаємодіяти з UI без використання Blueprint Widget?
8. Як додати подію для взаємодії через C++?
9. Що таке “UFUNCTION” і як його використовувати?
10. Як визначити відстань між персонажами через C++?
11. Як змінити поведінку персонажа через C++?
12. Як створити NPC через C++?
13. Як працює AI Controller у C++?
14. Як реалізувати переслідування гравця через C++?
15. Як засобами C++ забезпечити зміну поведінки NPC залежно від дій гравця?
16. Як передати інформацію між NPC та гравцем у C++?
17. Як засобами C++ змусити NPC взаємодіяти з предметами?
18. Як реалізувати розпізнавання NPC на відстані через C++?
19. Як працювати з “FVector” для визначення відстані між персонажами засобами C++?
20. Як створити власний компонент “Perception” для NPC у C++?
21. Як організовується передача стану NPC між клієнтом і сервером у C++?
22. Як організувати зміни емоцій NPC через C++?

23. Як засобами C++ створити власну систему станів для NPC?

24. Як засобами C++ забезпечити запам’ятовування NPC попередніх дій гравця?

25. Як працювати з “AnimInstance” у C++?

26. Як реалізувати групову взаємодію NPC через C++?

27. Як засобами C++ налаштувати NPC, який би змінював поведінку залежно від часу доби?

28. Як передавати команди між NPC через C++?

29. Як організувати зміну стану NPC через “Timers” у C++?

30. Як працювати з “Pathfinding” у C++?

Лабораторна робота №10
Створення системи бою з використанням C++
(4 бали)

Мета
Реалізувати систему бою між гравцем і ворогами.

Завдання
1. *(30% балів)* Реалізувати просту атаку (удар або постріл) з використанням C++.
2. *(40% балів)* Засобами C++ реалізувати систему нанесення ушкоджень: зменшення HP у гравця та NPC.
3. *(30% балів)* Додати візуальні ефекти атак через C++.

Хід виконання
1. Створення системи атаки:
   - додати клас C++ для атаки;
   - реалізувати логіку влучання по NPC через “Line Trace” або “Collision Sphere”;
   - встановити таймер на атаки для уникнення «спаму атак».
2. Реалізація системи ушкоджень:
   - створити окремий клас “HealthComponent” для управління HP;
   - реалізувати зменшення HP при атаці;
   - відобразити шкоду через анімацію або зміну кольору моделі.
3. Додавання ефектів атак:
   - використати “UParticleSystemComponent” для додавання частинок удару;
   - додати звукові ефекти при атаці;
   - перевірити правильність усіх анімацій атак.
4. Тестування:

- виконати серію тестових боїв між гравцем та NPC;
- перевірити коректність оновлення HP;
- проаналізувати ефективність ефектів атак.

Контрольні запитання

1. Як організовувати атаки персонажа через C++?
2. Що таке «Shield System» у C++?
3. Як реалізувати ближній бій на C++?
4. Як через C++ підключити ефекти до системи бою?
5. Як створити клас зброї в UE4 через C++?
6. Як працювати з “UAnimMontage” для анімацій атак у C++?
7. Як реалізувати ближній бій через “Collision Detection” у C++?
8. Як зберігати стан HP персонажа за допомогою C++?
9. Як створити клас “HealthComponent” у C++?
10. Як передавати інформацію про шкоду між об’єктами через C++?
11. Як працювати з “TakeDamage” у C++?
12. Як реалізувати систему броні персонажа через C++?
13. Як створити ефекти ударів за допомогою “UParticleSystemComponent”?
14. Як додати звук атаки персонажа через C++?
15. Як реалізувати дальній бій (стрільбу) у C++?
16. Як створити систему прицілювання через C++?
17. Як працює “LineTraceSingleByChannel” для перевірки попадань?
18. Як передати дані про атакуючого та ціль через C++?
19. Як реалізувати систему критичних ударів через C++?
20. Як реалізувати блокування атак через C++?
21. Як забезпечити правильне відображення шкоди у UI через C++?

22. Як налаштувати Knockback ефект після атаки через C++?
23. Як налаштувати реєстрацію атак у багатокористувацькому режимі через C++?
24. Як працює “Multicast RPC” у реалізації бойової системи?
25. Як уникнути дублювання нанесення шкоди через C++?
26. Як забезпечити взаємодію бойової механіки з AI NPC через C++?
27. Як створити власну логіку ухилення через C++?
28. Як забезпечити реакцію NPC на отримані удари через C++?
29. Як створити систему прокачування бойових умінь через C++?
30. Як забезпечити запис бойової статистики у грі через C++?

Лабораторна робота №11
Робота з багатокористувацьким середовищем в UE4
(4 бали)

Мета

Засвоїти основи створення багатокористувацьких ігор в UE4 засобами C++.

Завдання

1. *(30% балів)* Реалізувати підключення декількох гравців через C++.
2. *(40% балів)* Реалізувати реплікацію положення гравців та їхніх дій між клієнтами з використанням C++.
3. *(30% балів)* Додати систему чату між гравцями через C++.

Хід виконання

1. Створення мережевого GameMode:
    - додати клас C++ “AGameModeBase”;
    - реалізувати логіку підключення клієнтів до сервера;
    - переконатися, що клієнти можуть керувати своїми персонажами.
2. Реплікація даних:
    - реалізувати реплікацію руху гравців через “ACharacter”;
    - використати “ReplicatedUsing” для передачі змінних між клієнтами;
    - перевірити, що всі дії відображаються однаково у всіх гравців.
3. Створення системи чату:
    - реалізувати “RPC” виклики для відправлення повідомлень;

- створити клас “ChatManager” для обробки введених повідомлень;
- відобразити повідомлення в UI через “Slate UI” або “UMG”.

4. Тестування:
   - запустити сервер та підключити мінімум 2 клієнтів;
   - переконатися, що гравці бачать рухи один одного;
   - переконатися, що чат працює без затримок.

Контрольні запитання

1. Як створити мережевий GameMode в UE4 на C++?
2. Як працюють Remote Procedure Calls у C++?
3. Як розробити текстовий чат через C++ у багатокористувацькій грі?
4. Як створити серверну логіку у UE4 через C++?
5. Як підключити клієнтів до сервера через C++?
6. Як реалізувати реплікацію змінних у C++?
7. Як використовувати “APlayerController” для мережевої взаємодії?
8. Як створити систему автентифікації гравців через C++?
9. Як передавати команди між сервером і клієнтами через RPC?
10. Як реалізувати рух персонажів у мережевій грі через C++?
11. Як засобами C++ забезпечити відображення правильного стану гри для всіх клієнтів?
12. Як оптимізувати передачу даних у багатокористувацькому режимі через C++?
13. Як використовувати “Net Load on Client” для завантаження даних?

14. Як реалізувати синхронізацію анімацій між клієнтами через C++?
15. Як забезпечити захист від читерів у мережевій грі через C++?
16. Як реалізувати систему голосового чату через C++?
17. Як організувати логіку командної гри через C++?
18. Як зберігати статистику гравців на сервері через C++?
19. Як передавати повідомлення між гравцями через систему чату?
20. Як засобами C++ реалізувати систему команд у багатокористувацькому режимі?
21. Як засобами C++ реалізувати систему спостереження за іншими гравцями?
22. Як засобами C++ працювати з “ServerTravel” для зміни рівнів?
23. Як реалізувати мережевий режим глядача через C++?
24. Як реалізувати механіку затримки введення через C++?
25. Як організувати систему банів та блокування гравців через C++?
26. Як оптимізувати трафік у мережевій грі через C++?
27. Як створити систему матчмейкінгу через C++?
28. Як працює “NetDormancy” і коли його варто використовувати?
29. Як використовувати “FindSessions” для пошуку серверів?
30. Як реалізувати розгортання сервера в хмарі через C++?

Лабораторна робота №12
Збереження та завантаження ігрового процесу засобами C++
(4 бали)

Мета

Навчитися реалізовувати функції збереження та завантаження даних в UE4.

Завдання

1. *(40% балів)* Створити систему збереження стану гри через C++ (положення гравця, інвентар, статистика).
2. *(30% балів)* Реалізувати завантаження збереженого стану при запуску гри.
3. *(20% балів)* Реалізувати можливість ручного збереження через меню.
4. *(10% балів)* Перевірити, щоб збереження працювало у багатокористувацькому режимі.

Хід виконання

1. Система збереження:
   - створіть клас C++ для збереження даних (“Save Game Object”);
   - реалізуйте функцію збереження положення гравця, інвентаря та статистики.
2. Завантаження гри:
   - реалізуйте функцію завантаження збереженого стану при старті гри;
   - перевірте, чи коректно завантажуються дані.
3. Ручне збереження:
   - додайте опцію ручного збереження через меню;
   - налаштуйте багатослотову систему збереження.
4. Тестування:

- перевірте роботу збереження та завантаження у багатокористувацькому режимі.

Контрольні запитання

1. Що таке система збереження в UE4?
2. Як створити клас для збереження даних у C++?
3. Як зберегти положення гравця у грі засобами C++?
4. Як зберігати інвентар гравця засобами C++?
5. Як реалізувати збереження статистики гравця засобами C++?
6. Як створити систему автозбереження засобами C++?
7. Як реалізувати завантаження збережених даних засобами C++?
8. Як перевірити існування файлу збереження засобами C++?
9. Як організувати ручне збереження через меню та C++?
10. Як зберегти стан ворогів у грі засобами C++?
11. Як зберегти поточний прогрес рівня засобами C++?
12. Як налаштувати місце збереження файлів гри?
13. Як організувати резервні копії збережень?
14. Як реалізувати збереження у багатокористувацькому режимі?
15. Як перевірити коректність збережених даних?
16. Як організувати завантаження останнього збереження при запуску гри засобами C++?
17. Як зашифрувати файли збереження?
18. Як інтегрувати систему збереження через Blueprint і C++?
19. Як відновити дані у випадку пошкодження файлу збереження?
20. Як налаштувати систему багатослотового збереження у C++?
21. Як працює “UGameplayStatics::SaveGameToSlot”?

22. Як реалізувати асинхронне збереження у C++?
23. Як зберігати інформацію про стан анімацій NPC?
24. Як відстежувати зміни в ігровому світі та записувати їх у збереження?
25. Як зберігати інформацію про виконані місії у C++?
26. Як реалізувати систему чекпоінтів через C++?
27. Як забезпечити підтримку старих збережень після оновлення гри?
28. Як реалізувати очищення застарілих збережень автоматично?
29. Як працювати з “FArchive” для низькорівневого збереження?
30. Як створити систему “Quick Save” та “Quick Load”?

Лабораторна робота №13
Командне довгострокове завдання
(12 бали)

Мета

1. Розвинути навички роботи у команді.
2. Провести завершальні етапи розробки гри.
3. Розвинути навички представлення проєктів.

Завдання

1. *(25% балів)* Показати наповнення GitHub та Jira.
2. *(50% балів)* Представити розроблену гру.
3. *(25% балів)* На основі пройденого матеріалу сформуйте власне резюме засобами платформи worknuwm.

Хід виконання

1. Організація роботи:
   - продемонструйте засобами GitHub участь кожного члена команди у проєкті;
   - представте формулювання задач і графіки, які демонструють сумісну та системну роботу в Jira.
2. Розробка гри:
   - інтегруйте всі попередні наробки у єдиний додаток;
   - перевірте правильність роботи ключових функцій гри (мультиплеєр, система бою, збереження);
   - розгорніть гру.
3. Презентація проєкту:
   - підготуйте демонстрацію гри;
   - кожен член команди має пояснити свою частину роботи;
   - кожен член команди повинен вміти у загальних рисах пояснювати не свою частину роботи.

Оцінювання студентів відбуватиметься за критеріями згідно табл. 1.

Таблиця 1

Шкала оцінювання

№	Критерій	Вага	Оцінки			
			Студ1	Студ2	Студ3	Студ4
1	Організація роботи в Jira (Kanban-дошка, Story Points)	10%				
2	Використання GitHub (репозиторій, коміти, пул-реквести)	15%				
3	Якість візуального оформлення гри	15%				
4	Інтеграція ключових функцій (мультиплеєр, система бою, збереження)	15%				
5	Демонстрація гри та пояснення учасниками своїх частин	10%				
6	Додаткові творчі елементи (анімації, звуки, меню)	10%				
7	Якість сформованого резюме	25%				
Сума		100%				

Контрольні запитання

1. Які основні етапи розробки гри?
2. Як перевірити коректність роботи інтегрованих модулів у грі?
3. Що таке “Collision Detection”?
4. Як додати звукові ефекти у гру?
5. Як створити та налаштувати ігрове меню?
6. Які інструменти використовуються для командної роботи в UE4?
7. Як створити пул-реквест у GitHub?
8. Як вирішувати конфлікти у GitHub?

9. Як налаштувати логіку старту гри через меню?
10. Як забезпечити оптимізацію продуктивності гри?
11. Що включає в себе завершальний етап розробки гри?
12. Як правильно організувати структуру коду C++?
13. Як розгорнути гру?
14. Як документувати код та структуру проєкту для команди?
15. Як підготувати репозиторій GitHub для здачі проєкту?
16. Як забезпечити перевірку пам'яті для усунення можливих витоків?
17. Як забезпечити підтримку кількох мов через C++?
18. Як підготувати інтерактивний гайд для гравців?
19. Як організувати кінцеве тестування продуктивності гри?
20. Як оцінити відповідність проєкту поставленим цілям?
21. Як реалізувати розгортання гри на різних платформах?
22. Як підготувати гру до випуску в Steam або інших платформах?
23. Як організувати тестування навантаження?
24. Як налаштувати "Collision Detection"?
25. Як реалізувати систему хмарних збережень через?
26. Як організувати систему оновлень гри
27. Як забезпечити інтеграцію гри з соціальними мережами?
28. Як реалізувати динамічну зміну рівнів складності у грі?
29. Як забезпечити масштабованість гри при збільшенні кількості гравців?
30. Як забезпечити стабільну частоту кадрів у грі на різних пристроях?

Рекомендована література

Основна

1. Ровінський В. А. Кросплатформне програмування : навч. посіб. Івано-Франківськ : Прикарпатський національний університет імені Василя Стефаника, 2020. 151 с.

2. Костенко А. В., Костирко В. С., Плеша М.І . Кросплатформне програмування : навч. посіб. Львів : видавництво ЛТЕУ, 2019. 248 с.

3. Недашківський О. Л., Гусєва І. І. Кросплатформна розробка мобільних застосунків : навч. посіб. Київ : КПІ ім. Ігоря Сікорського, 2023. 221 с.

4. Nixon D. Beginning Unreal Game Development: Foundation for Simple to Complex Games Using Unreal Engine 4. 1st ed. New York : Apress, 2020. 413 p.

5. Костирко В. С., Охендушко Г. П. Розробка застосувань для мобільних пристроїв. Львів : Видавництво Львівського торговельно-економічного університету, 2023. 181 с.

6. Новоселов С. П., Сичова О. В. Основи розробки кросплатформного програмного забезпечення на Avalonia : навч. посіб. Харків : Видавництво Іванченка І. С., 2024. 267 с.

Додаткова

7. Лугова Т. А., Блажко О. А. Проєктування комп’ютерних ігор для навчання : навч. підруч. Одеса : ФОП «Побута», 2018. 212 с.

8. Eng L.Z. QT6 C++ GUI Programming Cookbook. 3rd ed. Birmingham : Packt Publishing, 2024. 449 p.

9. Agile-маніфест розробки програмного забезпечення. URL: https://agilemanifesto.org/iso/uk/manifesto.html (Last access: 11.12.2024).

Корисні посилання

10. From The University To Work. URL: https://worknuwm.com.ua/ (Last access: 11.12.2024).

11. Михайло Володимирович Бойчура - YouTube. URL: https://www.youtube.com/@mvboichura (Last accessed: 11.12.2024).

12. Освітні відео по Unreal Engine українською. Відеоуроки UE4. – YouTube. URL: https://www.youtube.com/playlist?list=PLWQWAXv3rt3E2auvNLo8WKhIeFYboaVIv (Last access: 11.12.2024).

13. Система контролю версій (Git) – YouTube. URL: https://www.youtube.com/playlist?list=PL9mn2EBC_SSyu6I4DQ9-r1vm_CX4jaPWf (Last access: 11.12.2024).

14. Jira Tools - YouTube. URL: https://www.youtube.com/playlist?list=PLqWD37Mj2te0xiDW-Q58ZIG7XZry-1-bk (Last accessed: 11.12.2024).

15. GitHub. URL: https://github.com/ (Last access: 11.12.2024).

16. Jira | Issue & Project Tracking Software | Atlassian. URL: https://www.atlassian.com/software/jira (Last access: 11.12.2024).

17. ChatGPT. URL: https://chatgpt.com/ (Last access: 11.12.2024).

18. Gemini. URL: https://gemini.google.com/ (Last access: 11.12.2024).

19. Groq is Fast AI Inference. URL: https://groq.com/ (Last access: 11.12.2024).

20. Chatbot Arena Leaderboard - a Hugging Face Space by lmarena-ai. URL: https://huggingface.co/spaces/lmarena-ai/chatbot-arena-leaderboard (Last access: 11.12.2024).