Finding AssignmExit

TEAM 6

Mingu kang
Jin yoo
Juyong shin
Eunseo jun

Table of Content

1. Proposal Explanation
2. Total Milestone Explanation
3. Detailed Milestone 1, 2 Explanation
4. Showing Game play video

LANGUAGES!!!!!!

UNREAL ENGINE'S C++
MAYA'S PYTHON

Project name

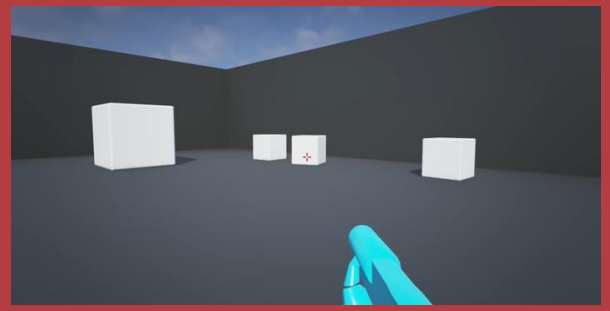
→ **Finding AssignmExit**

STORY

With the deadline just around the corner, the player have to find an exit to submit their assignments within the time limit…

MILESTONE

1. Realize first-person player.



2. Realize Maze map.

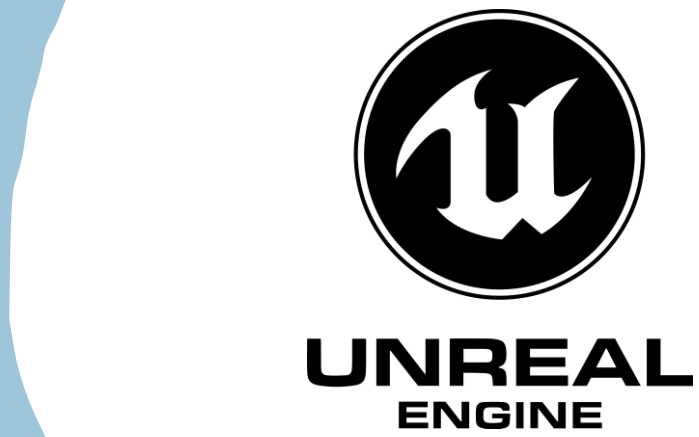
3. Realize Graphic
4. Realize Game Functions

11월 2일 - 오전 12시 30분 - 수업에서 줌으로 서로 팀 결정
11월 4일 - 오전 12시 30분- 성공적인 오픈소스 라이브러리를 발표
11월 5일 - 오후 1시반 줌 회의-(줌을 통해 앞으로 만들 프로젝트에 대해 각자 의견 내와서 상의)
11월 7일 - 10시반 줌- (저번 줌 회의 때 결정 난 미로게임을 세분화해서 테마 컨셉을 제안하고 투표해보여 구체적인 프로젝트 방향 설정)
11월 9일 - 오전 12시 30분 수업-(프로젝트 제안서를 작성해서 팀별로 발표했음)
11월 11일- 오전 12시 30분 수업-(프로젝트 일정 회의)
11월 12일 - 오전 12시- 내방역에서 만나서 마일스톤을 본격적으로 토의하고 정하며, 언리얼 엔진 설치 및 사용법 익히기 및 언리얼 엔진 c++코드를 이용한 미로게임 알고리즘 공부.
11월 15일 - 오후 11시- 줌을 통한 회의- 언리얼 엔진 c++코드 변경의 어려움을 느끼고 언리얼 엔진내의 블루 프린트와 블렌더, 유니티등 다른 프로그램을 활용하는 계획으로 변경
11월 16일 - 1.오전 12시- 만나 같이 수업을 듣고 오프라인으로 회의를 진행함
- 2. 블루프린트를 이용한 미로 건물 형성 시도(실패 후 피드백) - 마야가 파이썬코드를 이용한 점을 이용해 파이썬 코드를 이용해 원하는 크기의 미로를 생성해주는 알고리즘을 생성

11월 18일 - 오전 3시 반- 수업 수강후 그룹회의 진행- 플레이어 구현을 언리얼엔진 c++코드를 이용해 구현 성공.
11월 19일 - 오후 10시 반- 토론에서 파이썬코드(1차원 배열 인덱스 뽑는 코드)를 마야에 구현가능한 코드로 수정하여 마야로 미로 형성 구현 성공.
11월 21일 - 오후 2시 -지금까지 구현한 장면 정리 및 추가 작업할 내용 의논(마일스톤 3과 4에 해당되는 내용)
11월 22일 - 오후 9시 -중간 발표자료에 참고할 내용 의논 및 정리 와 발표대본에 대한 대략적인 핵심내용 점검
11월 23일- 오후 3시반 수업후- 발표PPT정리 및 발표대본 정리

Schedule

MILESTONE 1




```h
// Fill out your copyright notice in the Description page of Project Settings.

#pragma once

#include "CoreMinimal.h"
#include "GameFramework/Character.h"
#include "FPSCharacter.generated.h"

UCLASS()
class OSP_Y01_W01_API AFPSCharacter : public ACharacter
{
    GENERATED_BODY()

public:
    // Sets default values for this character's properties
    AFPSCharacter();

protected:
    // Called when the game starts or when spawned
    virtual void BeginPlay() override;

public:
    // Called every frame
    virtual void Tick(float DeltaTime) override;

    // Called to bind functionality to input
    virtual void SetupPlayerInputComponent(class UInputComponent* PlayerInputComponent) override;

    UFUNCTION()
        void MoveForward(float AxisValue);

    UFUNCTION()
        void MoveRight(float AxisValue);

    UFUNCTION()
        void StartJump();

    UFUNCTION()
        void StopJump();
};
```

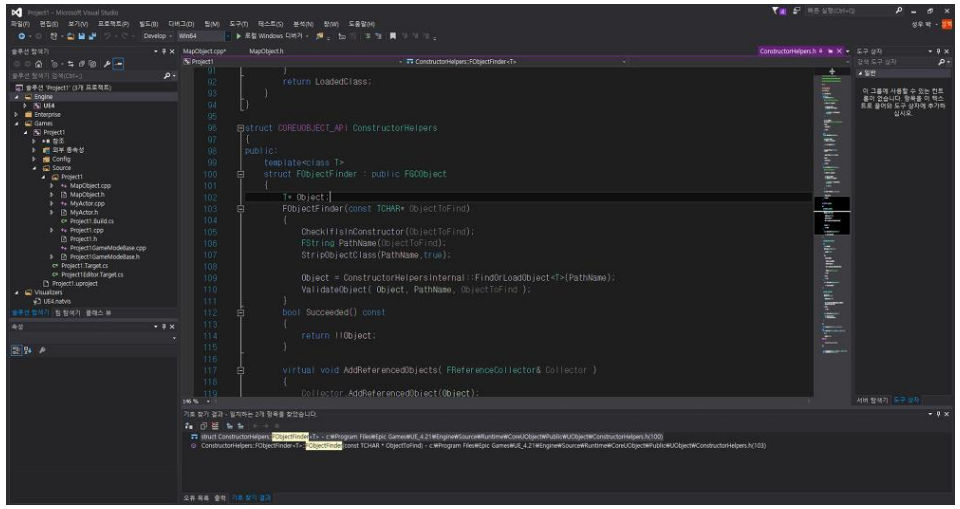
FPSCharacter.h

```cpp
void AFPSCharacter::Tick(float DeltaTime)
{
    Super::Tick(DeltaTime);

}

// Called to bind functionality to input
void AFPSCharacter::SetupPlayerInputComponent(UInputComponent* PlayerInputComponent)
{
    Super::SetupPlayerInputComponent(PlayerInputComponent);

    InputComponent->BindAxis("MoveForward", this, &AFPSCharacter::MoveForward);
    InputComponent->BindAxis("MoveRight", this, &AFPSCharacter::MoveRight);

    InputComponent->BindAxis("Turn", this, &AFPSCharacter::AddControllerYawInput);
    InputComponent->BindAxis("LookUp", this, &AFPSCharacter::AddControllerPitchInput);

    InputComponent->BindAction("Jump", IE_Pressed, this, &AFPSCharacter::StartJump);
    InputComponent->BindAction("Jump", IE_Released, this, &AFPSCharacter::StopJump);

}

void AFPSCharacter::MoveForward(float AxisValue)
{
    FVector Direction = FRotationMatrix(Controller->GetControlRotation()).GetScaledAxis(EAxis::X);
    AddMovementInput(Direction, AxisValue);
}

void AFPSCharacter::MoveRight(float AxisValue)
{
    FVector Direction = FRotationMatrix(Controller->GetControlRotation()).GetScaledAxis(EAxis::Y);
    AddMovementInput(Direction, AxisValue);
}

void AFPSCharacter::StartJump()
{
    bPressedJump = true;
}

void AFPSCharacter::StopJump()
{
    bPressedJump = false;
}
```

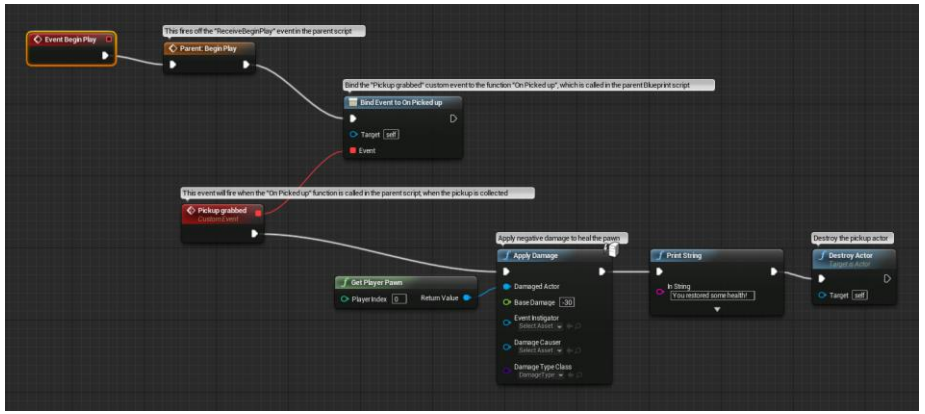
FPSCharacter.cpp

MILESTONE 2

UNREAL ENGINE C++



UNREAL ENGINE BLUE PRINT



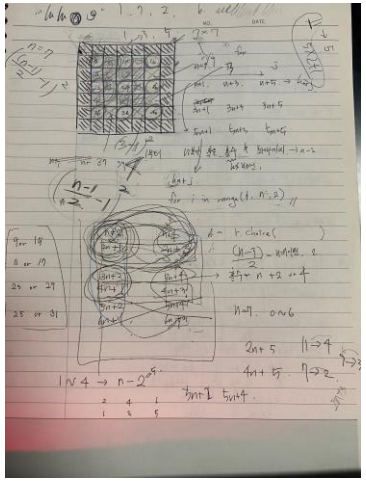
Coding is rarely used.

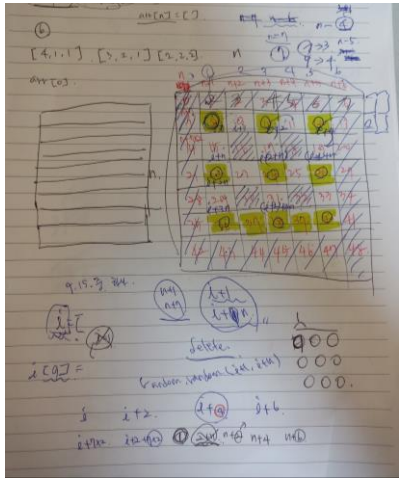
MILESTONE 2

BINARY TREE Algorithm

ELLER'S Algorithm

Recursive Backtracking Algorithm



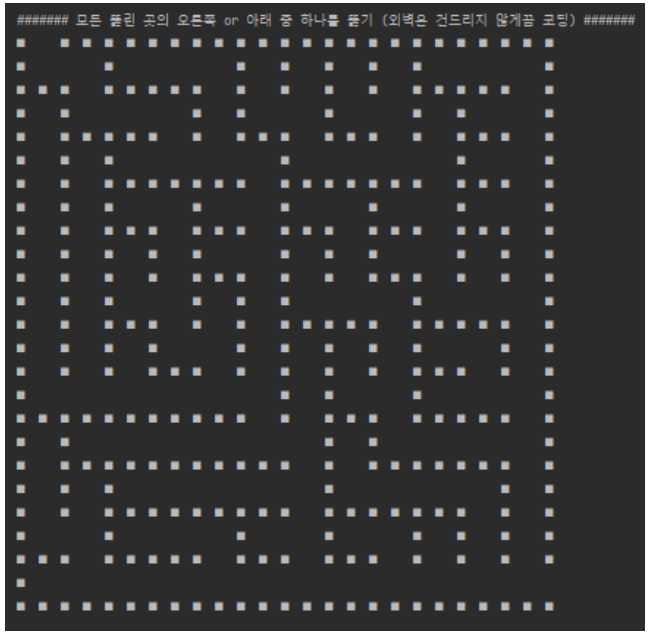


MILESTONE 2

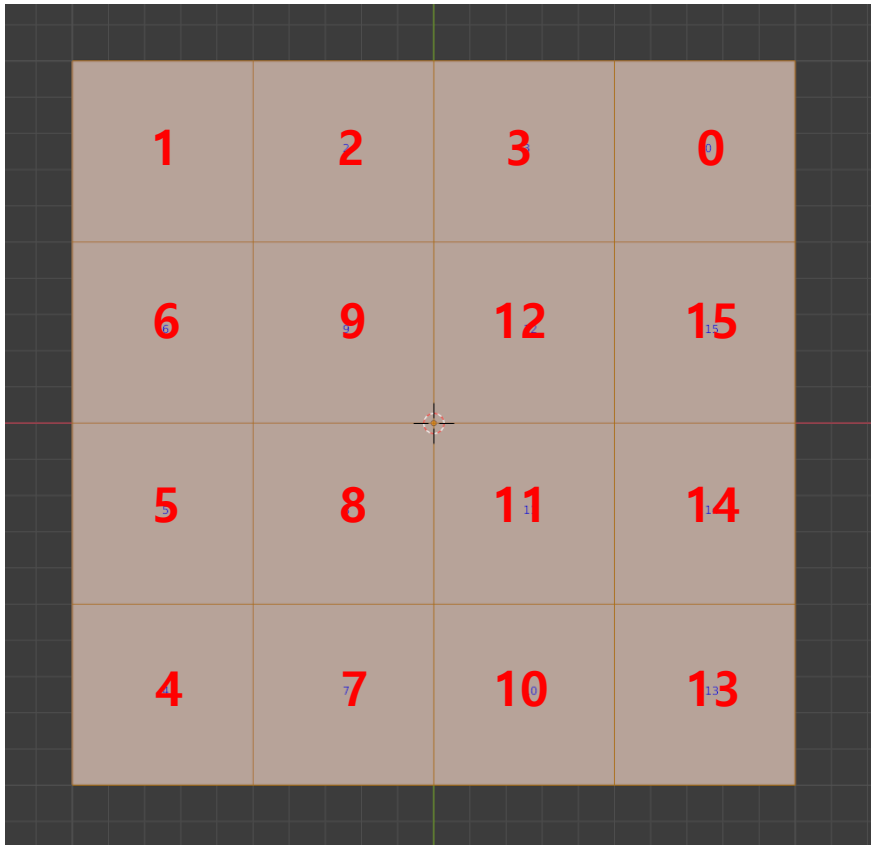
(Previsualization code)

```python
import random
n = int(input("미로의 크기 입력 : "))
list = [['■'] * n for i in range(n)]
######## N x N 맵 생성 ########
print("######## N x N 맵 생성 ########")
for i in range(n):
    for j in range(n):
        print(list[i][j], end=' ')
    print()
print()
######## 벽만 남기고 뚫기 ########
for i in range(1,n-1,1):
    for j in range(1,n-1,1):
        list[i][j] = ' '
######## 출력 ########
print("######## 벽만 남기고 뚫기 ########")
for i in range(n):
    for j in range(n):
        print(list[i][j], end=' ')
    print()
print()
######## 격자무늬로 벽 생성 ########
for i in range(1,n-1,1):
    for j in range(1,n-1,1):
        if i%2==0 or j%2==0:
            list[i][j] = '■'
```

```python
######## 출력 ########
print("######## 격자무늬로 벽 생성 ########")
for i in range(n):
    for j in range(n):
        print(list[i][j], end=' ')
    print()
print()
######## 모든 뚫린 곳의 오른쪽 or 아래 중 하나를 뚫기 (외벽은 건드리지 않게끔 코딩) ########
for i in range(1,n-1,2):
    for j in range(1,n-1,2):
        toss = random.choice([0,1])
        if toss==0:
            if (i+1 < n-1):
                list[i+1][j] = ' '
            else:
                list[i][j+1] = ' '
        else:
            if (j+1 < n-1):
                list[i][j + 1] = ' '
            else:
                list[i+1][j] = ' '
######## 시작지점 뚫기 ########
list[0][1] = ' '
######## 출력 ########
print("######## 모든 뚫린 곳의 오른쪽 or 아래 중 하나를 뚫기 (외벽은 건드리지 않게끔 코딩) ########")
for i in range(n):
    for j in range(n):
        print(list[i][j], end=' ')
    print()
print()
```



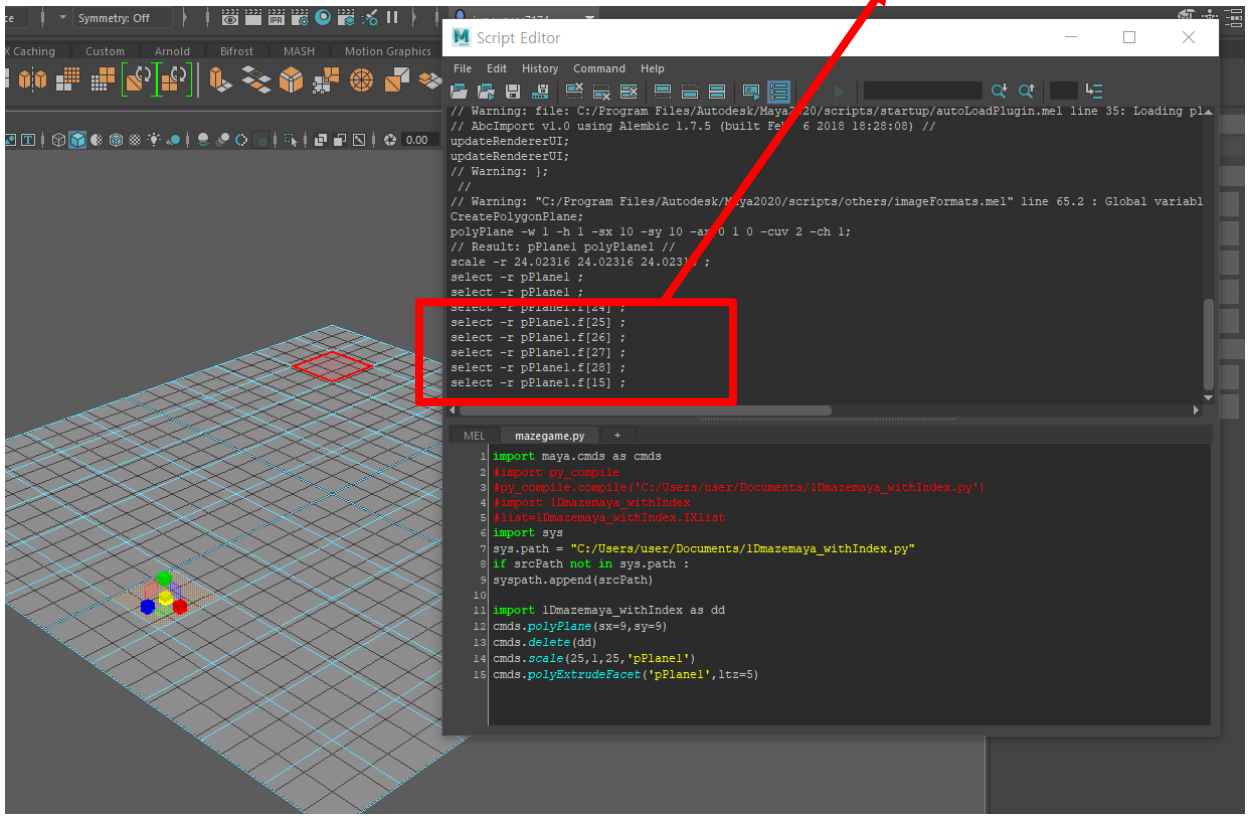
MILESTONE 2



MILESTONE 2

MAYA®

Typical index



MILESTONE 2

mazegame_maya.py - C:₩Users₩user₩Documents₩1-2학기 수업₩오픈소스 프...

File Edit Format Run Options Window Help

```
import maya.cmds as cmds
cmds.polyPlane(sx=9,sy=9)
#list = [1, 79, 10, 12, 14, 16, 28, 30, 32, 34, 46, 48, 50, 52, 64, 66, 68, 70,
cmds.delete('pPlane1.f[1]'
,'pPlane1.f[79]','pPlane1.f[10]','pPlane1.f[12]','pPlane1.f[14]',
'pPlane1.f[16]','pPlane1.f[28]','pPlane1.f[30]','pPlane1.f[32]'
,'pPlane1.f[34]','pPlane1.f[46]','pPlane1.f[48]',
'pPlane1.f[50]','pPlane1.f[52]','pPlane1.f[64]',
'pPlane1.f[66]','pPlane1.f[68]','pPlane1.f[70]',
'pPlane1.f[25]','pPlane1.f[43]','pPlane1.f[61]'
,'pPlane1.f[65]','pPlane1.f[67]','pPlane1.f[69]','pPlane1.f[11]'
,'pPlane1.f[13]','pPlane1.f[23]','pPlane1.f[37]',
'pPlane1.f[31]','pPlane1.f[41]','pPlane1.f[47]','pPlane1.f[57]'
,'pPlane1.f[51]')
cmds.scale(25,1,25,'pPlane1')
cmds.polyExtrudeFacet('pPlane1',ltz=20)
```

< MAYA'S PYTHON CODE >

1Dmazemaya.py - C:₩Users₩user₩Documents₩1-2학기 수업₩

File Edit Format Run Options Window Help

```
import random

arrf=[]#1차원 배열 미로 결과 인덱스

n = int(input("숫자 입력 : ₩n"))

arrf.append(1)
arrf.append(n*n-2)

for i in range(1, n, 2):
    for j in range(1, n, 2):
        arrf.append(i * n + j)

for i in range(2, n - 1, 2):
     arrf.append(i * n + n - 2)

for i in range(2, n - 1, 2):
    arrf.append((n - 2) * n + i)


list = []
for i in range(1, n - 2, 1):
    for j in range(2, n - 2, 2):
        if i % 2 == 0:
            list.append(i * n + j - 1)
        else:
            list.append(i * n + j)

arrs=[]
a = int((n-3)/2)
b = a*a*2
for i in range(0,b-a,1):
    if i%(a*2)<a:
        arrs.append(list[i])
        arrs.append(list[i + a])

arrrd = []
for i in range (0,b-1,1):
    if i%2==0:
        arrrd.append(arrs[i])
        arrrd.append(arrs[i+1])
        arrf.append(random.choice(arrrd))
        arrrd=[]

print(arrf)
```

< PYTHON CODE >

MILESTONE 2

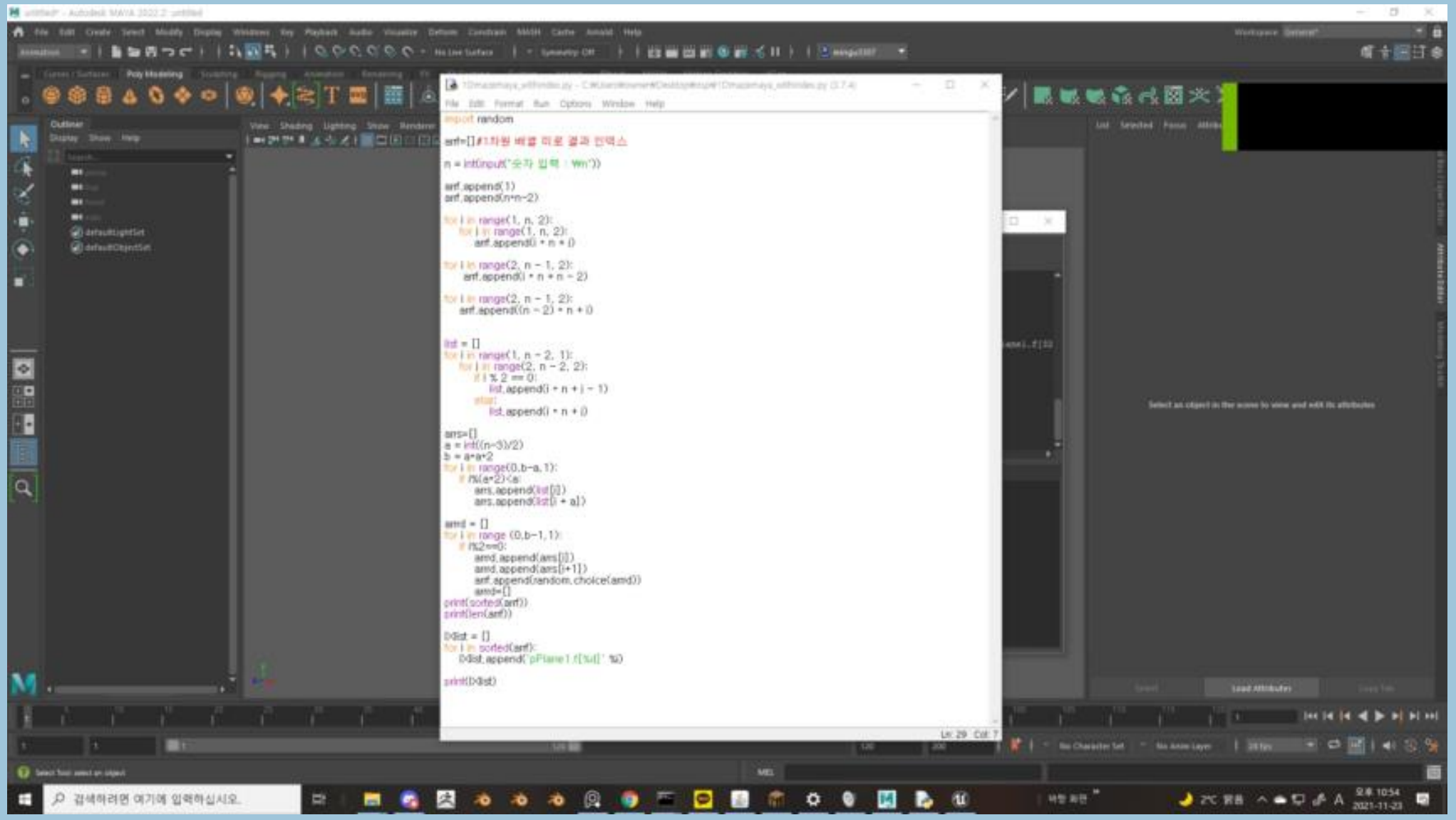
THE PROBLEM OF MAYA

1. One-dimensional arrangement.
2. Python Commend != Maya's python Commend
3. Maya cannot utilize variables. That's why I can't use the FOR statement.

Ex) for j in range(5):

cmds.delete('pPlane1.f[j]') -> impossibility

Game Play Video



☺ THANK YOU ☺

역할 분담

강민구 – 3d 미로를 위한 파이썬코드 작성, 플레이어 구현 코드 작성

유진 – 파이썬 마야지향 알고리즘 작성

신주용 – 블루프린트 미로 언리얼 공부 및 파이썬 코드 작성

전은서 – 마야 커맨드 창 파이썬 코드 미로 구현