

시스템 프로그래밍 HW3

[Curses 라이브러리를 이용한 Snake game]

프로젝트 주제 소개

- 이 프로젝트는 Curses 라이브러리를 사용하여 개발한 Snake 게임이다.
- Snake 게임은 뱀이 과일을 먹어 성장하거나, 벽에 부딪히면 게임이 종료되는 간단한 기능을 제공하는 게임이다.
- 사용자 터미널 창의 크기에 맞게 게임 화면이 자동 조정 되게끔 하였다.
- 사용자 입력을 받고, 사용자가 직접 화면에 뛰어 들 수 있게 하였다.
- 프로세스 간 역할을 분담하여 구현하였다.

기능 및 설계

1) 사용한 라이브러리 및 함수

- Curses 라이브러리

: 터미널 기반의 그래픽 환경을 구성하기 위해 해당 라이브러리를 사용하였다.

curses 모드 초기화, 화면 초기화, 특수 키 입력 처리, 그래픽 출력 등을 제어 하기 위해 curses 함수들을 사용하였다.

기능 및 설계

- `fork()`

: 자식 프로세스를 생성하기 위해 `fork()` 함수를 사용했다.

Snake 게임에서는 부모 프로세스가 게임 상태를 유지하고, 자식 프로세스가 사용자 입력 및 게임 로직 처리를 담당한다.

- `waitpid()`

: 부모 프로세스가 자식 프로세스의 종료를 감지하기 위해 `waitpid()` 함수를 사용했다.

부모 프로세스는 자식 프로세스 종료 여부를 확인하고, 게임이 종료된 후 화면을 정리한다

기능 및 설계

- noecho()

: 키 입력이 터미널 창에 나타나지 않게 하기 위해 사용했다.

- timeout(0)

: 입력 대기 시간 = 0 을 주어 키를 누르는 즉시 반응하도록 했다.

- sleep()

: game over 화면으로 전환 되었을 때, 화면이 유지되도록 하기 위해 사용했다.

기능 및 설계

2) 설계 단계

- Position 구조체

: 터미널 창의 좌표를 담아 뱀과 과일의 위치를 저장할 구조체 설계

- SharedData 구조체

: 부모 프로세스와 자식 프로세스가 공유할 데이터이며 뱀의 위치, 과일의 위치, 길이 및

점수를 저장하기 위한 구조체 설계

기능 및 설계

- 부모 프로세스의 역할

- a) 게임 화면 초기화 및 설정

- curses 라이브러리를 사용하여 터미널을 설정하고 초기화 한다.
 - 게임 화면의 크기를 얻어오고, 화면 크기 조정과 특수 키 입력 활성화 등이 포함된다.

- b) 뱀 / 과일 초기 위치 설정

- 초기에 뱀과 과일의 위치를 설정한다.
 - 자식 프로세스에 필요한 초기 정보들을 담은 구조체를 생성한다.

기능 및 설계

- 부모 프로세스의 역할

- c) 게임 루프 관리

- 자식 프로세스를 `fork()` 를 통해 생성한다.
 - 부모 프로세스는 일정 간격으로 화면을 갱신하고, 사용자 입력을 받아오면서 게임 루프를 유지한다.

- d) 자식 프로세스 종료 대기

- `waitpid` 함수를 사용하여 자식 프로세스의 종료를 감지한다.
 - 자식 프로세스의 종료를 감지하면, `curses` 라이브러리를 종료하고 프로그램을 종료한다.

기능 및 설계

- 자식 프로세스의 역할

- a) 게임 루프

- 무한 루프를 돌면서 게임 로직을 수행한다.
 - 키 입력 처리, 뱀의 움직임 업데이트, 충돌 검사, 화면 갱신 등

- b) 사용자 입력 처리

- getch() 함수를 사용하여 사용자의 키 입력을 받아, 원하는 방향으로 뱀 이동
 - 'q' 를 눌러 게임 종료

기능 및 설계

- 자식 프로세스의 역할

- c) 과일 먹기 및 스코어보드 갱신

- 뱀의 머리 좌표가 과일 좌표와 겹치면, 새로운 과일 위치를 무작위로 생성한다.
 - 추가로, 뱀의 길이를 늘리고 스코어보드에 점수를 증가시킨다.

- d) 충돌 검사

- 뱀이 벽과 충돌 했는지를 검사한다.
 - 충돌이 발생하면, 'Game over' 메시지와 함께 게임이 종료된다.

기능 및 설계

- 자식 프로세스의 역할

e) 화면 갱신

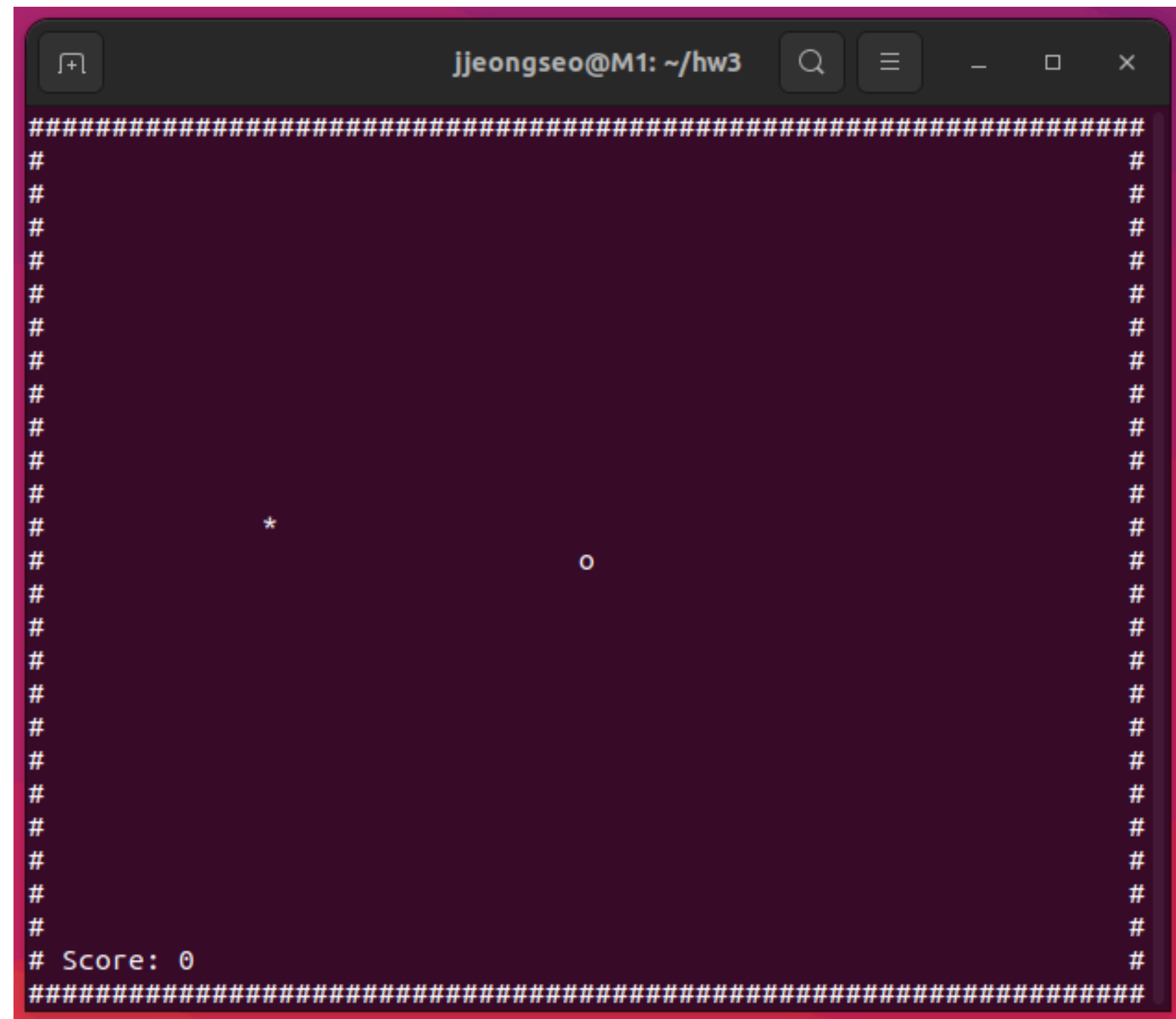
- 게임 상태에 따라 curses 라이브러리를 사용하여 화면을 갱신한다.

f) 게임 종료 및 부모 프로세스에 알림

- 게임이 종료되면, 자식 프로세스는 curses 모드를 종료하고, `exit()` 함수를 호출하여 부모 프로세스에게 종료되었음을 알린다.

수행 결과

(‘#’: 벽, ‘o’ : Snake, ‘*’ : Fruit)



```
#####
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
# Score: 0
#####
```

[게임 시작 직후 화면]

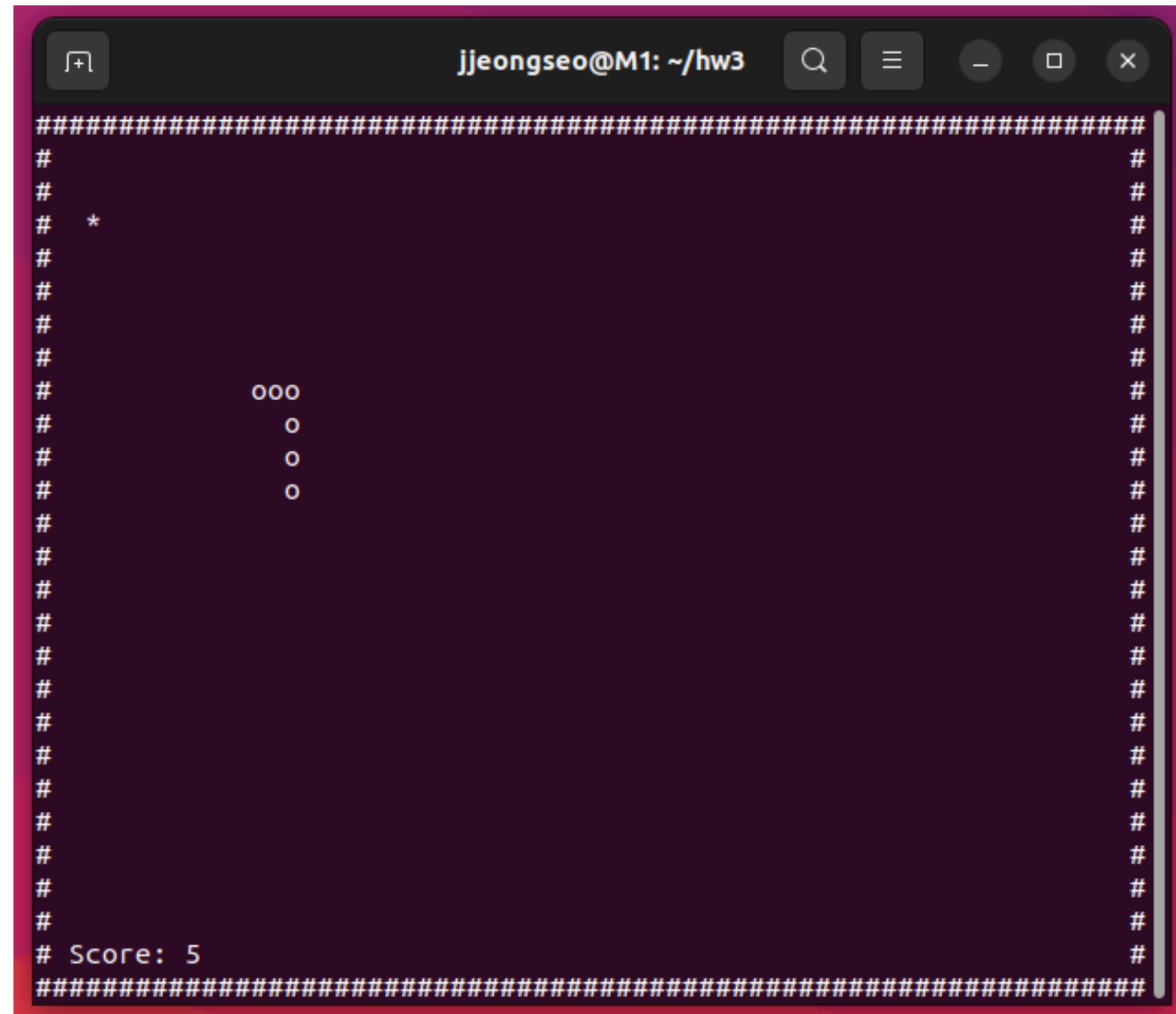


```
#####
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
# Score: 1
#####
```

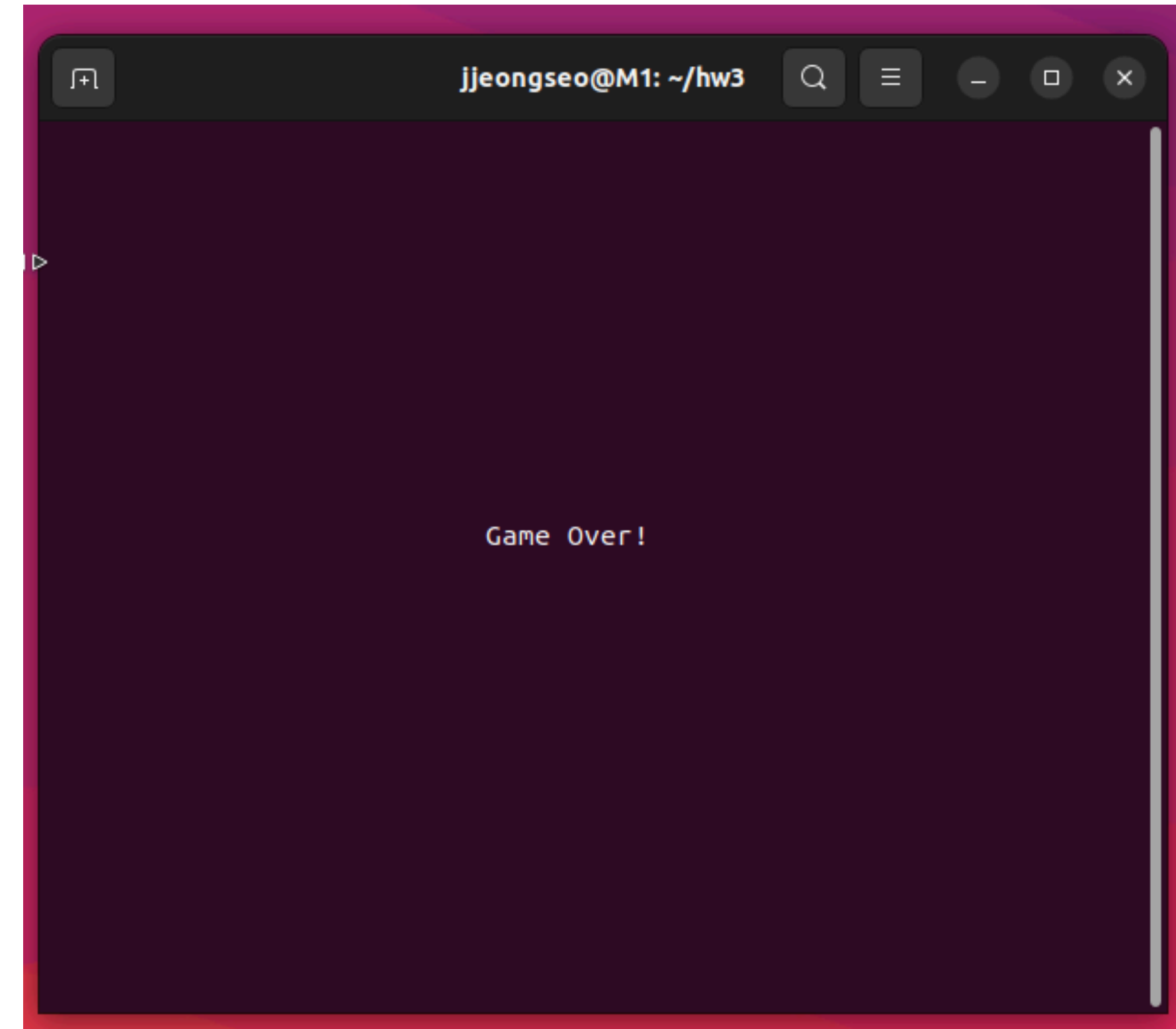
[과일 하나를 먹은 후]

수행 결과

(‘#’: 벽, ‘o’ : Snake, ‘*’ : Fruit)



[과일 4개를 먹은 후]



[충돌 후 게임 오버]

이슈 사항 및 해결

1) 키 입력 지연 이슈

- 이슈 : 키를 꾹 누르고 있는 경우, 입력 버퍼에 여러 개의 키 이벤트가 쌓이는 문제 발생
- 해결

: curses 라이브러리의 'timeout(0)' 함수를 통해 입력 대기 시간을 0으로 설정하여 즉시 키 입력을 받도록 하였다. 비교적 매끄럽게 키 입력을 받을 수 있으나 여전히 하나의 키를 꾹 누르는 경우, 어느 정도는 입력이 쌓이는 것을 확인하였다.

이슈 사항 및 해결

2) 게임 화면 이슈

- 이슈 1) 게임 오버 화면이 출력 되었을 때, 바로 프롬프트 창으로 넘어가는 현상
- 이슈 2) 화면이 깜빡이는 현상
- 해결

: 게임 오버 화면이 갱신 됨과 동시에 `usleep(5000000) = 5초` 를 주어서

게임 오버 화면이 일정 시간 유지 되도록 했다.

`clear()` 함수와 `refresh()` 함수의 적절한 배치로 이를 해결하였다.

이슈 사항 및 해결

3) 게임 종료 이후 화면 복구 이슈

- 이슈 1) 게임 종료 후에도 터미널 창이 올바르게 복구되지 않는 현상
- 이슈 2) 게임 종료 후 터미널 창이 즉시 종료되지 않고, 대기하는 현상
- 해결

: curses 모드에서 정상적으로 빠져나가지 않아 생긴 문제, 부모 프로세스 종료 이후에도 자식 프로세스의 종료를 확인 하지 않아 생긴 문제였다.

endwin() 함수로 curses 모드를 종료해주고, waitpid() 함수로 자식 프로세스의 종료를 확인 하면서 해결하였다.

Snake game 매뉴얼

[게임 가이드]

- 게임 시작

프로그램을 실행하면 바로 Snake game 이 시작된다.

- 이동

w / s / a / d 키를 사용하여 상하좌우로 Snake 를 이동 시킬 수 있다.

- 종료

게임 도중 'q' 키를 누르면 게임이 종료된다.

- Snake 와 Fruit 의 상호작용

뱀이 과일을 먹으면, 점수가 1점 올라가고 뱀의 길이가 1 증가한다.

컴파일 방법

[make 커맨드 이용]

- Make
: gcc -o hw03 hw03.c -lcurses 수행 (컴파일)
- Make clean
: rm -f hw03 수행 (실행파일 삭제)