



# 2023-2 고급프로그래밍

---

## 과제2: 짝꿍미게임(2)



한남대학교 컴퓨터공학과

## 과제2: 주꾸미 게임(2)



# 2-1) 게임데이터

---

- **과제1 코드를 수정해서**
  - player[] → 구조체배열로 변경(다음 슬라이드 참고)
  - jjuggumi.dat에서 데이터를 읽어온 후, 정상적으로 동작하도록 수정한다.
- **플레이어 (기본)스탯**
  - 지능, 힘, 스테미나
  - 스테미나는 모두 100%로 시작(0%~100%)
  - 각 플레이어는 장비 1개씩 소지 가능
    - 장비를 장착하면 3종 스탯에 버프 추가
- **유효 스탯: 이벤트 발생 시 실제로 적용되는 스탯(실수)**
  - 유효지능 = 지능 x 스테미나(%)
  - 유효힘 = 힘 x 스테미나(%)
    - 스테미나가 100%면 x1.0
    - 50%면 x0.5

# jjuggumi.h

<jjuggumi.dat> 수정

- 플레이어 정보(0명인 경우는 없음)

- bool player[] → PLAYER player[] 로 수정

```
typedef struct {  
    int id;  
    char name[100];  
  
    // 능력치: 지능, 힘, 스테미나  
    int intel, str, stamina;  
  
    // 현재 상태  
    bool is_alive; // 탈락했으면 false  
    bool hasitem; // 아이템이 있으면 true  
    ITEM item;     // 아이템 1개 장착 가능  
} PLAYER;  
  
PLAYER player[PLAYER_MAX]; // 구조체 배열
```

10

과제1에서 player[]에 해당되는 값

load

6						인원수: 6명
공유	8	7				이름, 지능, 힘
강새벽	9	4				
성기훈	7	6				
조상우	10	5				
한미녀	6	4				
오일남	9	3				
10						
볼펜	1	0	0			
커터칼	-1	3	0			
안경	3	0	0			
운동화	0	1	20			
방탄복	0	-2	60			
헬멧	0	-1	3			
야구방망이	-1	5	0			
탄산음료	0	0	1			
빵	1	1	2			
홍삼	1	1	5			

# jjuggumi.h

- 아이템 정보

- 아이템 개수
- 각 아이템의
  - 이름
  - 지능 버프, 힘 버프, 스테미나 버프

```
// 아이템을 장착하면 스탯 증가
typedef struct {
    char name[100];
    int intel_buf, str_buf, stamina_buf;
} ITEM;
```

```
ITEM item[ITEM_MAX];
```

10

load

<jjuggumi.dat>

```
6
공유 8 7
강새벽 9 4
성기훈 7 6
조상우 10 5
한미녀 6 4
오일남 9 3
10
볼펜 1 0 0
커터칼 -1 3 0
안경 3 0 0
운동화 0 1 20
방탄복 0 -2 60
헬멧 0 -1 3
야구방망이 -1 5 0
탄산음료 0 0 1
빵 1 1 2
홍삼 1 1 5
```

개수  
이름, 스탯보정

# jjuggumi.c

- **jjuggumi\_init()**

- 파일에 저장된 데이터를 읽어서 구조체 배열에 저장하는 함수
- 전역변수
  - n\_player, n\_item

- **메인함에서 호출**

- -1을 리턴하면 “file open error”를 출력하고 프로그램 종료

```
int jjuggumi_init() {
    srand((unsigned int)time(NULL));

    FILE* fp;
    fopen_s(&fp, DATA_FILE, "r");
    if (fp == NULL) {
        return -1; // -1 리턴하면 메인함수에서 처리하고 종료
    }

    // 플레이어 데이터 load
    fscanf_s(fp, "%d", &n_player);
    for (int i = 0; i < n_player; i++) {
        // 아직 안 배운 문법(구조체 포인터, 간접참조연산자)
        PLAYER* p = &player[i];

        // 파일에서 각 스탯 최대값 읽기
        fscanf_s(fp, "%s%d%d",
                  p->name, (unsigned int)sizeof(p->name),
                  &(p->intel), &(p->str));
        p->stamina = 100; // 100%

        // 현재 상태
        p->is_alive = true;
        p->hasitem = false;
    }

    // 아이템 데이터 load
    fscanf_s(fp, "%d", &n_item);
    for (int i = 0; i < n_item; i++) {
        fscanf_s(fp, "%s%d%d%d",
                  item[i].name, (unsigned int)sizeof(item[i].name),
                  &(item[i].intel_buf),
                  &(item[i].str_buf),
                  &(item[i].stamina_buf));
    }

    fclose(fp);
    return 0;
}
```

“jjuggumi.dat”

# canvas.c

- 스탯이 화면에 표시되도록 한다.

- 기본스탯(+아이템버프)
- 예)

...

no. of players left: 3

	intl	str	stm
player 0: alive	5(+0)	3(+0)	100%
player 3: alive	2(+0)	6(+0)	50%
player 8: alive	1(+0)	9(+0)	10%

<참고>

유효지능	유효힘
5.0	3.0
1.0	3.0
0.1	0.1

- 스태미나

- 게임을 한 개 마칠 때마다 일정량 회복(적당히 밸런스 조절하기 - 4~50%?)
- 특정 행동을 할 때 소모됨
- 아이템은 장착하는 시점에 스태미나 회복(100%는 넘을 수 없음)

# 게임 순서

---

- 무궁화 꽃이 피었습니다.
- 야간운동
  - 숙소에서 아이템을 두고 벌어지는 쟁탈전
- 줄다리기
  - 패자부활전
- 제비뽑기
  - 우승자 1명 뽑기
  - (과제1 -> 과제2)



## 2-2) 야간운동

- 게임마스터는 필드에 아이템과 플레이어를 랜덤하게 뿌린다.
  - 아이템은 'I'로 표시, 겹치지 않게 배치
  - 어떤 아이템인지는 획득할 때가 아니라 배치할 때 결정할 것
  - 예) 아이템 2개, 플레이어 3명 배치

```
*****
*      2      I      *
*                               *
*              I      *
*                3      *
*                               *
*      0      *
*****
```

- 운동 규칙
  - 사용자는 0번 플레이어를 방향키로 조종
    - 0번이 탈락하면 자동 진행
  - 나머지 플레이어들은
    - 아이템과 (아이템을 가진 플레이어)들 중, 가장 가까운 곳을 향해서 이동
    - 일정한 주기로 이동. 주기는 적당히 맞추기
  - 각 플레이어는 아이템 1개씩 획득 가능
  - 'q'를 누르면 종료

- 
- **상호작용: 플레이어가 아이템 칸에 이동**
    - (또는 아이템과 인접한 칸으로 이동 - 각자 편한 방법으로 구현)
  - **아이템이 없는 상태면**
    - 아이템을 획득
  - **아이템이 있으면**
    - 장비한 아이템과 땅에 떨어진 아이템을 교환
    - 플레이어 0은 교환할 건지 지나칠 건지 키보드로 입력해서 선택
      - 구현 방법은 자유
    - 나머지 플레이어는 50% 확률로 무시하거나 교환함
  - 상호작용 결과를 다이얼로그, 또는 가운데 출력 창에 잠깐동안 표시

---

- 
- **상호작용: 플레이어끼리 만났을 때(인접 칸에 위치)**

- 0번 플레이어가 다른 플레이어와 만났을 때
  - 게임을 잠시 멈추고 선택지를 고름
- 다른 플레이어끼리 만났을 때,
  - 두 명 중 아이템이 없는 플레이어가 랜덤하게 선택지를 골라서 행동
  - (둘 다 아이템이 있거나 둘 다 없으면 둘 중 아무나 한명이 행동함)
  - 상호작용 과정과 결과를 다이얼로그, 또는 가운데 출력 창에 잠깐동안 표시

- **선택지**

- **\*\*각 플레이어는 다른 플레이어의 상태를 알 수 없음**
- 1) **강탈 시도**: 유효 힘을 비교해서 더 높다면 성공
  - 성공하면
    - 아이템 강탈 또는 (시도하는 플레이어가 이미 아이템이 있으면)교환
    - 성공하면 스테미나 40%, 실패하면 60% 소모
    - 스테미나가 부족해도 현재 스테미나가 0%보다 크기만 하면 시도 가능
- 2) **회유 시도**: 유효 지능을 비교해서 더 높다면 성공
  - 성공하면
    - 아이템 강탈 또는 (시도하는 플레이어가 이미 아이템이 있으면)교환
    - 성공하면 스테미나 20%, 실패하면 40% 소모
    - 스테미나가 부족해도 현재 스테미나가 0%보다 크기만 하면 시도 가능
- 3) **무시**: 그냥 지나감

# 게임 순서

---

- 무궁화 꽃이 피었습니다.
- 야간운동
  - 한밤중 숙소에서...
- 줄다리기
  - 패자부활전
- 제비뽑기
  - 과제1 -> 과제2

# 2-3) 줄다리기

## • 패자부활전

- 모든 플레이어가 참가
- 탈락했던 플레이어는
  - (아이템은 없는 상태로 시작)
  - 줄다리를 이기면 부활, 지면 탈락
- 탈락하지 않았던 플레이어가 패배하면
  - 아이템 몰수
  - 힘과 지능 기본값이 절반으로 하락

## • 배치

- 3x? 충분한 길이의 맵 생성
- 0번, 2번 행 가운데는 구멍
- 1번 행 가운데 3칸은 줄(---)
- 플레이어들을
  - 가운데를 기준으로 번갈아 왼쪽, 오른쪽으로 배치

```
#####  
#          6420---1357          #  
#####
```

```
str:      1
```

```
no. of players left: 8
```

```
player 0: alive
```

```
player 1: alive
```

```
player 2: alive
```

```
player 3: alive*
```

```
player 4: alive
```

```
player 5: alive*
```

```
player 6: alive
```

```
player 7: alive
```

탈락했던 플레이어

- **str: 유효 힘의 합: (-왼쪽 팀) + (오른쪽 팀)**

- 예)

- 플레이어 6, 4, 2, 0의 유효 힘의 합이 25.
- 플레이어 1, 3, 5, 7의 유효 힘의 합이 26.
- →  $\text{str} = (-25.0) + (26.0) = 1.0$

- 1초마다

- str이 음수면 왼쪽, 양수면 오른쪽으로
- 한 칸씩 당겨짐
- 구멍에 빠진 플레이어는 탈락
  - 다이얼로그로 탈락자 표시

```
#####  
#          6420---1357          #  
#####
```

str: 1.0

```
no. of players left: 8  
player 0: alive  
player 1: alive  
player 2: alive  
player 3: alive*  
player 4: alive  
player 5: alive*  
player 6: alive  
player 7: alive
```

- 예)

- 1초 후

str>0이므로 오른쪽으로 한 칸 당겨짐



```
#####  
#          6420---1357          #  
#####
```

str: 1.0

- 2초 후

- 다이얼로그 출력(0번 플레이어 탈락)

str>0이므로 오른쪽으로 한 칸 당겨짐



```
#####  
#          642---1357          #  
#####
```

str: 1.0

- 2초~3초 사이

- str에서 탈락한 0번 플레이어의  
• 유효 힘을 제외

- 예) 0번 플레이어의 유효 힘이 0.4라면 2초 이후 str의 값은 1.4

## • 기본 조작키(게임마스터)

- 'q': 줄다리기 종료
- 'z': 왼쪽으로 당김(str 1 감소)
- '/': 오른쪽으로 당김(str 1 증가)

## • 예)

- 이전 슬라이드에서 2~3초 사이에
- 'z'를 연타하면 연타 횟수만큼 str 감소
- 3초가 됐을 때
  - str이 음수이므로 줄은 왼쪽으로 한 칸
  - 당겨진 후
  - str은 'z'나 '/'를 누르지 않은 상태로
  - 돌아간다.
  - (키 입력으로 변한 값은 줄이 이동한
  - 후에는 초기화됨)

```
#####  
#           642---1357           #  
#####
```

str: 1.4 -> 0.4 -> -0.6 -> ...



```
#####  
#           642---1357           #  
#####
```

str: -1.6

```
#####  
#           642---1357           #  
#####
```

str: 1.4



---

## • 눅기

- 'x'를 누르면 왼쪽 팀이, '.'를 누르면 오른쪽 팀이 **눅기**를 사용한다.
- 다이얼로그 또는 중간 출력창에 표시
- **눅기**를 쓴 팀은 순간적으로 힘이 두 배가 되고,
  - 그 턴에는 줄이 1칸이 아니라 2칸을 움직인다.
  - 두 팀이 함께 눅기를 쓸 수도 있다.
- 단, 눅기를 쓴 팀의 모든 플레이어는 스태미나가 30%씩 감소한다.
  - 0% 미만으로는 내려가지 않음

# 과제2 제출 방법

---

- **11월 30일(목요일)까지**
  - delay: 1일 10%, 최대 5일
- **hi-class 고급프로그래밍(실) → 시험 및 설문 → 과제2 꾸꾸미게임(2) 제출**
  - 팀 이름, 팀원 이름, github 주소
  - private으로 작업, 제출일+5일 이후에 공개
  - 제출 시점: 최종수정일 기준
  - 제출 전 윈도우+VS에서 잘 실행되는지 반드시 확인
  - \*\*과제1과 같은 repo를 쓰지 말고 새로운 repo 에서 작업하기