

수업명 C 프로그래밍 및 실습

HANG MAN GAME

최종 보고서

제출일자: 20231224

제출자명: 신승환

제출자학번: 192280

1. 프로젝트 목표

1) 배경 및 필요성

C 프로그램을 이용해 만들 수 있는 미니 게임을 고민하다가 행맨을 생각해냈다.

2) 프로젝트 목표

미리 정해진 단어 중 하나를 선택하고 사용자가 그 단어를 맞추도록 하는 것이다.

3) 차별점

기존의 행맨 게임은 맞출 때까지 할 수 있었다면 시도 할 수 있는 횟수를 미리 정해 놓고 게임을 시작하기 때문에 오히려 더 재밌게 진행 할 수 있다. 거기에 더해 랜덤 위치의 문자만 보여줌으로써 게임의 흥미를 더 높인다

2. 기능 계획

1) 기능 1 (단어 목록 정의 및 무작위 선택)

- 설명 : 2 차원 배열(단어의 개수, 단어의 최대 길이) 사용할 단어 목록을 저장한다. 그리고 무작위 함수 RAND 를 이용하여 이중에서 무작위로 단어를 선택한다

2) 기능 2 (행맨 게임 함수)

- 설명 : 정해진 횟수 안에서 사용자가 한 단어씩 입력하면서 정해진 단어를 추측한다.

기능 1) 행맨 게임 함수에 필요한 단어 초기화 + 랜덤 위치의 문자만 보여주기

기능 2) 반복문을 통해 사용자가 횟수 내에 맞추거나 종료될때까지 반복

기능 3)글자 추측 및 확인

기능 4) 게임 종료 메시지를 출력 – 정해진 횟수 안에 실패 했는지 아닌지

3) 기능 3 (메인 함수, 게임 재시작)

기능 1) 위에서 만든 행맨 게임함수를 가지고 메인 함수를 작성

기능 2) 사용자가 게임을 재시작 하거나 종료를 선택할 수 있게 함

기능 3) 헤더파일을 추가

3. 기능 구현

(1) 기능 1(단어 목록 정의 및 무작위 선택)

- 입출력 : 없음
- 설명 : 2 차원 배열(단어의 개수, 단어의 최대 길이) 사용할 단어 목록을 저장한다. 그리고 무작위 함수 RAND 를 이용하여 이중에서 무작위로 단어를 선택한다.
- 적용된 배운 내용 : 2 차원 배열, 포인터, 랜덤함수
- 코드 스크린샷 :

```
//단어의 개수와 최대길이를 2차원 배열로 하는 words에 단어 목록 저장
char words[NUM_WORDS][MAX_WORD_LENGTH] = {
    "apple",
    "banana",
    "cherry",
    "grape",
    "orange"
};

//문자열 포인터를 반환하는 함수를 생성 -> 반환된 포인터는 선택된 무작위 단어의 첫번째 문자를 가리킴
char* getRandomWord() {
    int index = rand() % NUM_WORDS; //0부터 NUM_WORDS - 1 사이의 무작위 정수를 생성
    return words[index];
}
```

(2) 세부기능 1(행맨 게임 함수에 필요한 단어 초기화)

- 입출력 : 없음
- 설명 : 행맨 게임 함수에 필요한 변수들을 적절하게 선언 및 초기화
- 적용된 배운 내용 : 변수 선언, 배열, 포인터
- 코드 스크린샷 :

```

void playHangman() {
    //게임에 필요한 변수 선언 및 초기화
    char* wordToGuess = getRandomWord();
    int wordLength = strlen(wordToGuess);
    char visibleWord[MAX_WORD_LENGTH];
    int numTries = 0;
    int numCorrectGuesses = 0;
}

```

(3) 세부기능 2(랜덤 위치의 문자만 보여주기)

- 입출력 : 없음
- 설명 : 선택된 단어의 절반 정도를 visibleWord 배열에 복사하고 나머지 절반은 '_'로 채워서 플레이어에게 보여줘서 어느 정도의 문자만 보여준다.
- 적용된 배운 내용 : 배열, 반복문, 조건문
- 코드 스크린샷

```

// 절반의 문자만 보이도록 설정
for (int i = 0; i < wordLength; i++) {
    visibleWord[i] = (i < wordLength / 2) ? wordToGuess[i] : '_';
}
visibleWord[wordLength] = '\0';

printf("행맨 게임을 시작합니다. 단어를 맞춰보세요!\n");

```

(4) 세부기능 3(글자 추측 및 확인)

- 입출력 : 추측된 단어를 입력받아 guess에 넣는다.
- 설명 : 플레이어는 단어를 맞추기 위해 여러 번 시도할 수 있으며, 게임은 플레이어가 모든 글자를 맞추거나 시도 횟수를 모두 사용할 때까지 계속된다.
- 적용된 배운 내용 : scanf_s, while 반복문, 포인터, for 반복문, if 조건문
- 코드 스크린샷 :

```

//단어 상태를 플레이어에게 보여줌
while (numTries < MAX_TRIES && numCorrectGuesses < wordLength) {
    printf("단어: %s\n", visibleWord);
    printf("추측한 단어를 입력하세요: ");

    //플레이어에게 한 글자 추측받음
    char guess;
    scanf("%c", &guess);

    //추측한 글자가 단어에 있는지 반복문과 조건문을 통해 확인
    int found = 0;
    for (int i = 0; i < wordLength; i++) {
        if (wordToGuess[i] == guess && visibleWord[i] == '_') {
            visibleWord[i] = guess;
            found = 1;
            numCorrectGuesses++;
        }
    }

    //추측 결과에 따라서 플레이어에게 힌트 제공
    if (!found) {
        numTries++;
        printf("틀렸습니다. 남은 시도 횟수: %d\n", MAX_TRIES - numTries);
    }
    else {
        printf("맞췄습니다!\n");
    }
}

```

(5) 세부기능 4(게임 종료 메시지를 출력 - 정해진 횟수 안에 실패 했는지 아닌지)

- 입출력 : 없음
- 설명 : 플레이어가 게임에서 성공했는지 실패했는지에 따라 적절한 피드백 제공. 성공시엔 느 축하메시지를, 실패 시에는 아쉬움을 나타내며 정답 공개
- 적용된 배운 내용 : if 문과 while 문
- 코드 스크린샷 :

```

//게임 종료 메시지 출력
if (numCorrectGuesses == wordLength) {
    printf("축하합니다! 정답은 W"%sW" 였습니다!\n", wordToGuess);
}
else {
    printf("아쉽습니다. 정답은 W"%sW" 였습니다.\n", wordToGuess);
}

```

(6) 기능3 - 세부기능 1,2(메인함수 및 게임 재시작)

- 입출력 : 플레이어가 1을 입력하면 재시작을 0을 입력하면 게임 종료
- 설명 : 플레이어는 게임을 연속해서 여러 번 플레이할 수 있으며, 언제든지 게임을 종료할

수 있습니다

- 적용된 배운 내용 : 반복문, 조건문

- 코드 스크린샷 :

```
int main() {
    srand(time(NULL));

    int playAgain = 1;
    do {
        playHangman();

        // 게임 종료 후, 사용자에게 다시 플레이할지 묻기
        printf("게임을 다시 플레이하시겠습니까? (1:재시작, 0:종료): ");
        while (scanf_s("%d", &playAgain) != 1) {
            while (getchar() != '\n'); // 입력 버퍼 비우기
            printf("잘못된 입력입니다. 다시 입력해주세요 (1:재시작, 0:종료): ");
        }

        // 입력 값에 따라 반복 또는 종료
        if (playAgain == 1) {
            printf("\n새 게임을 시작합니다.\n\n");
        }
        else if (playAgain == 0) {
            printf("게임을 종료합니다. 안녕히 가세요!\n");
        }
    } while (playAgain);

    return 0;
}
```

(7) 세부기능 3(헤더파일 이용)

- 입출력 : 없음

- 설명 : 헤더파일을 이용해 더 핵심 기능을 하는 코드들만 소스파일에 작성가능하게 이용한다.

- 적용된 배운 내용 : 헤더파일

- 코드 스크린샷 :

```
main.c  X  player.c  player.h
Project4  (전역 범위)  main()
1  #include "player.h"
2
3  int main() {
4      srand(time(NULL));
5
6      int playAgain;
7      do {
8          playHangman();
9
10         printf("게임을 다시 플레이하시겠습니까? (1:재시작, 0:종료): ");
11         while (scanf("%d", &playAgain) != 1) {
12             while (getchar() != '\n');
13             printf("잘못된 입력입니다. 다시 입력해주세요 (1:재시작, 0:종료): ");
14         }
15
16         if (playAgain == 1) {
17             printf("\n새 게임을 시작합니다.\n\n");
18         }
19         else if (playAgain == 0) {
20             printf("게임을 종료합니다. 안녕히 가세요!\n\n");
21         }
22     } while (playAgain);
23
24     return 0;
25 }
```

```
main.c  X  player.c  X  player.h
Project4  (전역 범위)  getRandomWord()
1  #include "player.h"
2
3  #define NUM_WORDS 5
4  #define MAX_WORD_LENGTH 10
5  #define MAX_TRIES 6
6
7  char words[NUM_WORDS][MAX_WORD_LENGTH] = {
8      "apple",
9      "banana",
10     "cherry",
11     "grape",
12     "orange"
13 };
14
15 char* getRandomWord() {
16     int index = rand() % NUM_WORDS;
17     return words[index];
18 }
19
20 void playHangman() {
21     //게임에 필요한 변수 선언 및 초기화
22     char* wordToGuess = getRandomWord();
23     int wordLength = strlen(wordToGuess);
24     char visibleWord[MAX_WORD_LENGTH];
25     int numTries = 0;
26     int numCorrectGuesses = 0;
27
28     // 절반의 문자만 보이도록 설정
29     for (int i = 0; i < wordLength; i++) {
30         visibleWord[i] = (i < wordLength / 2) ? wordToGuess[i] : '_';
31     }
32     visibleWord[wordLength] = '\0';
```

```
main.c  player.c  player.h  (전역 범위)
Project4
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4  #include <time.h>
5
6  // 무작위 단어를 반환하는 함수
7  char* getRandomWord();
8
9  // 행맨 게임을 실행하는 함수
10 void playHangman();
11
```

4. 테스트 결과

(1) 기능 1 (단어 목록 정의 및 무작위 선택)

- 설명 : 배열 만들기
- 테스트 결과 스크린샷 : 없음

(2) 세부기능 1) 행맨 게임 함수에 필요한 단어 초기화

- 설명 : 변수 선언 및 초기화만 진행함
- 테스트 결과 스크린샷 : 없음

(3) 세부기능 2) 랜덤 위치의 문자만 보여주기

- 설명 : 랜덤의 위치만 보여주기 위해 설정함
- 테스트 결과 스크린샷 :

C:\Users\W192280\Desktop\Project3\Wx64\Debug\Project3.exe

행맨 게임을 시작합니다. 단어를 맞춰보세요!

단어: ap_____

추측한 단어를 입력하세요:

(4) 세부기능 3) 글자 추측 및 확인

- 설명 : 글자를 추측하여 플레이어가 입력한 글자를 확인하는 과정 반복

- 테스트 결과 스크린샷 :

C:\Users\W192280\Desktop\Project3\Wx64\Debug\Project3.exe

행맨 게임을 시작합니다. 단어를 맞춰보세요!

단어: ap_____

추측한 단어를 입력하세요: aplekk

틀렸습니다. 남은 시도 횟수: 5

단어: ap_____

추측한 단어를 입력하세요:

(5) 세부기능 4) 게임 종료 메시지를 출력 – 정해진 횟수 안에 실패 했는지 아닌지

-설명 : 정답인지 실패인지 확인하여 플레이어에게 알려줌

-테스트 결과 스크린샷 :

단어: ban_____

추측한 단어를 입력하세요:

sf

틀렸습니다. 남은 시도 횟수: 0

아쉽습니다. 정답은 "banana" 였습니다.

게임을 다시 플레이하시겠습니까? (1:재시작, 0:종료): 0

게임을 종료합니다. 안녕히 가세요!

(6) 기능 3 - 세부기능 1,2 (메인함수 및 게임 재시작)

- 설명 : 글자를 추측하여 플레이어가 입력한 글자를 확인하는 과정 반복

- 테스트 결과 스크린샷 :

```
단어: ap____
추측한 단어를 입력하세요: af
틀렸습니다. 남은 시도 횟수: 0
아쉽습니다. 정답은 "apple" 였습니다.
```

(7) 세부기능 3 (헤더 파일 이용)

- 설명 : 헤더파일을 이용한 코드로 변경
- 테스트 결과 스크린샷 : 테스트 결과의 차이는 없음, 코드 구성만 바뀜

5. 계획 대비 변경 사항

1) 기능 2 의 세부기능 1

- 이전 : 행맨 게임 함수에 필요한 단어 초기화
- 이후 : 행맨 게임 함수에 필요한 단어 초기화 + 랜덤 위치의 문자만 보여주기
- 사유 : 기존의 행맨 게임에 흥미와 난이도를 높임으로써 플레이하는 사용자가 더 재밌게 게임을 할 수 있게 만들기 위해

2)기능 3 의 세부기능 3 추가

- 이전 : 없었음
- 이후 : 전체 파일을 헤더파일을 이용하여 나눈다
- 사유 : 프로젝트 진행중에 있어서 수업 시간에 헤더파일을 배웠고 헤더파일을 이용하였을 때 핵심 기능을 하는 코드들만 소스파일에 작성가능하고 특정 기능을 하는 함수, 구조체들의 재사용이 쉬워진다는 장점이 있다

6. 느낀점

처음 시작할 때는 어떤 것으로 해야할 지부터 막막했으나 점점 구체화시켜가는 과정이 흥미롭게 다가왔던 거 같고 배울수록 어려운부분이 늘어나는 점에 대해 더 많이 해봐야겠다는 생각을 하게 되었다.