프로그래밍 1 및 실습

영문 타자 연습 프로그램

분반	공통반	
담당 교수님	문영성 교수님	
제출마감	- 소스코드: 2018.06.05 - 출력물 : 2018.06.06	
소속	컴퓨터학부	
구성원	홍상원	20142577
	김연수	20162347
	김희원	20172577

1. 소스코드

```
#include <stdio.h>
#include <stdbool.h>
#include <stdlib.h>
#include <unistd.h>
#include <termios.h>
#include <time.h>
#include <string.h>
#include <sys/time.h>

//메뉴 개수
#define NO 5
//낱말 연습 단어 개수
#define WORD_SIZE 100
//짧은 글 연습 문장 개수
#define SHORT_SIZE 30
//긴 글 연습 글 개수
#define LONG_SIZE 4
//입력 값 최대 길이
#define MAX_SIZE 1000

//ESC ASCII 값
#define ESC 0x1B
//백스페이스 ASCII 값
#define DEL 0x7F
```

```
//낱말 연습 입력 횟수
#define WORD_ROUND 20
//짧은 글 연습 입력 횟수
#define SHORT_ROUND 5
//긴 글 연습 페이지 수
#define NUM_OF_PAGES 2
//긴 글 연습 한 페이지 행의 개수
#define NUM_OF_ROWS 5

//긴 글 연습 글 저장을 위한 Article 타입 정의
typedef struct _Article {
        char *page1[NUM_OF_ROWS];
        char *page2[NUM_OF_ROWS];
} Article;

void clear(void);
void move_cursor(int x, int y);
int getch(void);
void start_msg(int no);
int menu(void);

int cal_accuracy_for_short(char *str1, char *str2, int last);
int cal_speed_for_short(char *input, char *sen, int idx, struct timeval *start, struct timeval *end);
int cal_accuracy_for_long(char input_buf[][NUM_OF_ROWS][MAX_SIZE], Article *article, int page_num,
int x_pos, int y_pos);
int cal_speed_for_long(char input_buf[][NUM_OF_ROWS][MAX_SIZE], Article *article, int page_num,
```

```
int x_pos, int y_pos, struct timeval *start, struct timeval *end);

void exercise_pos(void);
void exercise_word(void);
void exercise_short(void);
void exercise_long(void);
void end(void);
void error(void);

int main(void) {
	int no;

	while(1) {
		no = menu();

		switch(no) {
		case 1://자리 연습
			exercise_pos();
			break;
		case 2://낱말 연습
			exercise_word();
			break;
		case 3://짧은 글 연습
			exercise_short();
			break;
		case 4://긴 글 연습
```

```
                        exercise_long();
                        break;
                case 5://프로그램 종료
                        end();
                default://1 ~ 5 이외의 값 입력 시 오류
                        error();
                }
        }

        return 0;
}

//화면을 clear하는 함수
void clear(void) {
        system("clear");
}

//(x, y) 좌표로 화면애서 커서를 이동시키는 함수
void move_cursor(int x, int y) {
        printf("₩e[%d;%df", y, x);
}

//한 문자를 입력받는 함수, 입력한 문자가 화면에 보이지 않는다.
int getch(void) {
    struct termios oldattr, newattr;
    int ch;
```

```
	tcgetattr(STDIN_FILENO, &oldattr);
	newattr = oldattr;
	newattr.c_lflag &= ~(ICANON|ECHO);
	tcsetattr(STDIN_FILENO, TCSANOW, &newattr);
	ch = getchar();
	tcsetattr(STDIN_FILENO, TCSANOW, &oldattr);
	return ch;
}

//시작 메시지를 출력하는 함수
void start_msg(int no) {
		char *msg[NO] = {"₩b₩b₩b", "자리 연습", "낱말 연습", "짧은 글 연습", "긴 글 연습"};
		clear();
		move_cursor(0, 0);
		printf(">> 영문 타자 연습 프로그램 : %s <<₩n", msg[no]);
}

//메뉴를 출력하고 메뉴 번호를 입력받는 함수
int menu(void) {
		int no;
		start_msg(0);
		printf("1. 자리 연습₩t");
		printf("2. 낱말 연습₩n");
		printf("3. 짧은 글 연습₩t");
		printf("4. 긴 글 연습₩n");
		printf("5. 프로그램 종료₩n₩n");
```

```
        printf("번호를 선택하세요: ");
        scanf("%d", &no);
        /*
         * scanf() 함수는 입력값을 받고 엔터를 치면 변환 명세에 해당하는 값을 변수(no)에 저장한다.
         * 그러나 입력 버퍼에 '₩n'이 남아 있기 때문에 다음에 곧바로 문자를 입력받게 되면 '₩n'이 영향을 줄 수 있다.
         * 따라서 getchar()로 입력을 받아서 '₩n'을 입력받아서 입력 버퍼에서 빼낸다.
         */
        getchar();
        return no;
}

//자리 연습을 실행시키는 함수
void exercise_pos(void) {
        //정확히 입력한 단어 개수
    int right = 0;
    //pro : 진행도, err : 오타수, acc : 정확도
        int pro = 0, err = 0, acc = 0;
        //입력 받은 문자
    int input;
        //a ~ z, A ~ Z 중에 입력해야 할 문자
        char ch;
        //대문자라면 true(1), 소문자라면 false(0)
    bool is_uppercase;
        //ch와 input이 다르면 true, 같으면 false
```

```
bool is_err = false;

//랜덤 숫자의 seed를 현재 시간으로 설정
srand(time(NULL));

//올바른 값 20개를 입력할 때까지 반복
while(right < 20) {
	start_msg(1);
	move_cursor(0, 2);
	printf("진행도 : %3d%%₩t오타수 : %3d₩t정확도 : %3d%%₩n", pro, err, acc);

	//올바른 값을 입력했을 때만 다음 라운드로 이동한다.
	if(!is_err) {
		//소문자라면 0, 대문자라면 1
		is_uppercase = rand() % 2;
		if (!is_uppercase)//소문자일 때
			ch = rand() % 26 + 'a';//소문자 랜덤
		else// 대문자일 때
			ch = rand() % 26 + 'A';//대문자 랜덤
	}

	move_cursor(0, 5);
	printf("%c", ch);

	move_cursor(0, 6);
	input = getch();
```

```
		//ESC를 입력하면 메뉴로 복귀
		if(input == ESC)
			return;

		printf("%c", input);
		//올바른 값을 입력했을 경우
		if(input == ch) {
			++right;//정확한 값 입력 개수 증가
			pro += 5;//진행도 5% 증가
			is_err = false;
		}
		else {//올바르지 않은 값을 입력했을 경우
			++err;//에러 입력 개수 증가
			is_err = true;
		}
		//정확도 계산
		acc = right * 100 / (right + err);
}

//최종 결과
start_msg(1);
move_cursor(0,2);
printf("진행도 : %3d%%₩t오타수 : %3d₩t정확도 : %3d%%₩n", pro, err, acc);
move_cursor(0, 5);
printf("종료하려면 ENTER를 누르세요.");
```

```
		move_cursor(0, 6);

		//엔터를 입력하면 메뉴로 복귀
		while(getch() != '₩n');
}

//낱말 연습을 실행시키는 함수
void exercise_word(void) {
		//낱말 데이터 100개
		char *word[WORD_SIZE] = {
			"appointment", "communication", "doubt", "remind", "phenomenon",
			"meanwhile", "import", "interacrion", "likewise", "logger",
			"munufacture", "maximize", "diminish", "renew", "physical",
			"content", "extend", "ingredient", "gaze", "regular",
			"minimize", "content", "transform", "intend", "expend",
			"official", "special", "combine", "reputation", "devote",
			"gradual", "therefore", "class", "indeed", "require",
			"comprehend", "debate", "announce", "structure", "indicate",
			"resist", "exit", "resource", "function", "tough",
			"postpone", "outcome", "undergo", "preceive", "circular",
			"hospital", "outstand", "expose", "enforce", "biography",
			"concern", "circumstance", "isolation", "analyze", "commute",
			"solve", "abstraction", "state", "common", "character",
			"last", "perhaps", "matter", "privilege", "iron",
			"vanity", "fatigue", "calamity", "virtue", "impulse",
			"hyprocrisy", "system", "monotony", "conquest", "insight",
```

```
            "influence", "fortitude", "compromise", "prosperity", "comfort",
            "symphony", "candidate", "homage", "conceit", "frustration",
            "impression", "identity", "eloquence","adolescence", "heritage",
              "add", "vivid", "simultaneous", "acquaintance", "ready"
};
//word 인덱스
int no;
//올바른 낱말을 입력한 횟수
int match = 0;
//진행도, 오타수, 정확도
int progress = 0, typos = 0, accuracy = 0;
//입력 받은 문자
int input;
//입력 받은 낱말
char input_buf[MAX_SIZE];
//input_buf 인덱스
int idx = 0;
//반복문을 위해 사용
int i, j;
//입력받은 낱말 중복 체크용 배열
int selected_word[WORD_ROUND];
//중복 체크를 위한 플래그, 중복이면 true, 중복이 아니면 false
bool duplicate;

//랜덤 숫자의 seed를 현재 시간으로 설정
srand(time(NULL));
```

```
//20번 반복하여 낱말 입력
for(i = 0; i < WORD_ROUND; i++) {
        start_msg(2);
        duplicate = false;
        while(1) {
                //word 인덱스를 랜덤으로 구하기
                no = random() % WORD_SIZE;
                //이미 나왔던 낱말인지 중복 체크
                for(j = 0; j < i; j++) {
                        if(selected_word[j] == no) {
                                duplicate = true;
                                break;
                        }
                        else
                                duplicate = false;
                }
                //중복된 낱말이 아닐 때까지 반복
                if(!duplicate)
                        break;
        }
        //한 번 나온 낱말을 중복 체크용 배열에 저장
        selected_word[i] = no;

        memset(input_buf, 0x00, MAX_SIZE);
```

```
		move_cursor(0,2);
		printf("진행도 : %3d%%₩t오타수 : %3d₩t 정확도 : %d%%₩n", progress, typos,
accuracy);
		move_cursor(0, 5);
		printf("%s", word[no]);

		move_cursor(0,7);
		while (1) {
			input = getchar();
			//엔터를 입력받았을 때
			if(input == '₩n') {
				input_buf[idx] = 0;
				//###를 입력받은 경우 메뉴로 복귀
				if(!strcmp("###", input_buf))
					return;
				//잘못된 낱말을 입력한 경우
				if(strcmp(word[no], input_buf))
					typos++;//오타수 증가
				else//올바른 낱말을 입력한 경우
					match++;//매칭수 증가

				//정확도 계산
				accuracy = match * 100 / (typos + match);
				//진행도 5% 증가
				progress += 5;
				//input_buf 인젝스 초기화
```

```
                        idx = 0;
                        break;
                    }
                    else//엔터 이외의 문자를 입력받은 경우 input_buf에 저장
                        input_buf[idx++] = (char) input;
                }
            }

            //최종 결과
            start_msg(2);
            move_cursor(0,2);
            printf("진행도 : %3d%%₩t오타수 : %3d₩t 정확도 : %3d%%₩n", progress, typos, accuracy);
            move_cursor(0, 5);
            printf("종료하려면 ENTER를 누르세요.");
            move_cursor(0,7);

            //엔터를 입력받으면 메뉴로 복귀
            while(getch() != '₩n');
}

//짧은 글 연습에서 정확도를 계산하는 함수
int cal_accuracy_for_short(char *input, char *sen, int idx) {
            //정확한 문자 개수
            int correct_count = 0;
            //입력해야 할 문장 길이
            int len = strlen(sen);
```

```
	//결과값
	int result;
	//반복문을 위해 사용
	int i;
	//체크해야할 데이터의 끝 위치
	int last;
	//입력 받은 문자가 없을 경우 정확도는 0%
	if(idx == 0)
		return 0;

	//입력받은 문장이 입력해야 할 문장보다 긴 경우 넘치는 부분 무시
	last = idx <= len ? idx : len;
	//정확한 문자 개수 세기
	for(i = 0; i < last; i++) {
		if(input[i] == sen[i])
			correct_count++;
	}

	//정확도 계산
	result = correct_count * 100 / last;
	return result;
}

//짧은 글 연습에서 현재 타수를 계산하는 함수
int cal_speed_for_short(char *input, char *sen, int idx, struct timeval *start, struct timeval *end) {
	//시작 시간부터 입력받은 시간까지의 기간
```

```
long long seconds = end->tv_sec - start->tv_sec;
//현재 타자
int speed;
//반복문을 위해 사용
int i;
//정확한 문자 개수
int correct_count = 0;
//체크해야할 데이터의 끝 위치
int last;
//입력해야 할 문장 길이
int len = strlen(sen);

//시작한 상황이면 현재 타자수 0
if(seconds == 0)
	return 0;

//입력받은 문장이 입력해야 할 문장보다 긴 경우 넘치는 부분 무시
last = idx <= len ? idx : len;
//정확한 문자 개수 세기
for(i = 0; i < last; i++) {
	if(input[i] == sen[i])
		correct_count++;
}

//현재 타수 계산
speed = (correct_count * 60) / seconds;
```

```
	return speed;
}

//짧은 글 연습을 실행시키는 함수
void exercise_short(void) {
	//문장 30개
	char *sentence[SHORT_SIZE] = {
		"The grass is always greener on the other side of the fence.",//0
		"Don't judge a book by its cover.",//1
		"Strike while the iron is hot.",//2
		"Too many cooks spoil the broth.",//3
		"You can't have your cake and eat it too.",//4
		"Many hands make light work.",//5
		"When in Rome, do as the Romans do.",//6
		"Don't cross the bridge until you come to it.",//7
		"Honesty is the best policy.",//8
		"Practice makes perfect.",//9
		"Where there's a will, there's a way.",//10
		"Look before you leap.",//11
		"Beggars can't be choosers.",//12
		"Don't make a mountain out of an anthill.",//13
		"An apple a day keeps the doctor away.",//14
		"The early bird catches the worm.",//15
		"Better late than never.",//16
		"The cat is out of the bag.",//17
		"Two wrongs don't make a right.",//18
```

```
        "Always put your best foot forward.",//19
        "Rome wasn't built in a day.",//20
        "It's better to be safe than sorry.",//21
        "Don't bite the hand that feeds you.",//22
        "The squeaky wheel gets the grease.",//23
        "Don't bite off more than you can chew.",//24
        "You made your bed, now you have to lie in it.",//25
        "Actions speak louder than words.",//26
        "It takes two to tango.",//27
        "Don't count your chickens before they hatch.",//28
        "It's no use crying over spilled milk."//29
};
//sentence 인덱스
int no;
//진행도, 현재 타수, 최고 타수, 정확도
int progress = 0, current_speed = 0, max_speed = 0, accuracy = 0;
//입력 받은 문자
int input;
//입력 받은 문장
char input_buf[MAX_SIZE];
//input_buf 인덱스
int idx = 0;
//반복문을 위해 사용
int i, j;
//중복 체크 플래그
bool duplicate;
```

```
//한 번 나왔던 sentence 인덱스
int selected_short[SHORT_ROUND];
//시작 시간을 알기 위한 플래그
bool is_first = true;
//시작, 끝 시간
struct timeval start, end;

//랜덤 숫자의 seed를 현재 시간으로 설정
srand(time(NULL));

//5번 반복하여 문장 입력
for(i = 0; i < SHORT_ROUND; i++) {
        duplicate = false;
        while(1) {
                //sentence 인덱스 랜덤으로 구하기
                no = random() % SHORT_SIZE;
                //중복 체크
                for(j = 0; j < i; j++) {
                        if(selected_short[j] == no) {
                                duplicate = true;
                                break;
                        }
                        else
                                duplicate = false;
                }
                if(!duplicate)
```

```
                        break;
                }
                //한 번 나온 sentence의 인덱스 저장
                selected_short[i] = no;

                memset(input_buf, 0x00, MAX_SIZE);
                while(1) {
                        start_msg(3);
                        move_cursor(0, 2);
                        printf("진행도 : %3d%%₩t현재타수 : %3d₩t최고타수 : %3d₩t정확
도 : %3d%%₩n", progress, current_speed, max_speed, accuracy);
                        move_cursor(0, 5);
                        printf("%s", sentence[no]);

                        move_cursor(idx, 7);
                        input = getch();
                        //ESC 입력 시 메뉴로 복귀
                        if(input == ESC)
                                return;
                        else if(input == DEL) {//백스페이스 입력
                                if(idx <= 0) {//지울 문자가 없을 경우
                                        continue;
                                }
                                //인덱스를 줄여 문자 지우기
                                idx--;
                                //정확도 계산
```

```
                    accuracy = cal_accuracy_for_short(input_buf, sentence[no], idx);
                    //현재 타수 계산
                    current_speed = cal_speed_for_short(input_buf, sentence[no], idx,
&start, &end);
                    //최고 타수 계산
                    max_speed = max_speed < current_speed ? current_speed :
max_speed;
                }
                else if(input == '₩n') {//엔터 입력 시
                    //진행도 20% 증가
                    progress += 20;
                    memset(input_buf, 0x00, MAX_SIZE);
                    //input_buf 인덱스 초기화
                    idx = 0;
                    //현재 타수 초기화
                    current_speed = 0;
                    //정확도 초기화
                    accuracy = 0;
                    //시작 시간 재설정을 위해 플래그를 true로 초기화
                    is_first = true;
                    break;
                }
                else {
                    //처음 입력 시
                    if(is_first) {
                        //시작 시간 구하기
```

```
                    gettimeofday(&start, NULL);
                    is_first = false;
                }
                else//현재까지 시간 구하기
                    gettimeofday(&end, NULL);
                input_buf[idx++] = input;
                //정확도 구하기
                accuracy = cal_accuracy_for_short(input_buf, sentence[no], idx);
                //현재 타수 계산
                current_speed = cal_speed_for_short(input_buf, sentence[no], idx,
&start, &end);
                //최고 타수 계산
                max_speed = max_speed < current_speed ? current_speed :
max_speed;
            }

            move_cursor(0, 7);
            input_buf[idx] = 0;
            printf("%s", input_buf);
        }
    }

    //최종 결과
    start_msg(3);
    move_cursor(0, 2);
    printf("진행도 : %3d%%₩t현재타수 : %3d₩t최고타수 : %3d₩t정확도 : %3d%%₩n",
```

```
progress, current_speed, max_speed, accuracy);
        move_cursor(0, 5);
        printf("종료하려면 ESC를 누르세요.");
        move_cursor(0, 7);

        //ESC를 입력하면 메뉴로 복귀
        while(getch() != ESC);
}

//긴 글 연습에서 정확도를 계산하는 함수
int cal_accuracy_for_long(char input_buf[][NUM_OF_ROWS][MAX_SIZE], Article *article, int page_num,
int x_pos, int y_pos) {
        //반복문을 위해 사용
        int i, j;
        //정확히 입력한 문자 개수
        int correct_count = 0;
        //문장 길이
        int len;
        //결과값
        int result;
        //입력한 문자 개수
        int total = 0;

        //두 번째 페이지일 경우
        if(page_num == 1) {
                //첫 번째 페이지에서 올바르게 입력한 문자수 세기
```

```
		for(i = 0; i < 5; i++) {
			len = strlen(article->page1[i]);
			for(j = 0; j < len; j++) {
				total++;
				if(input_buf[0][i][j] == article->page1[i][j])
					correct_count++;
			}
		}

		//두 번째 페이지에서 올바르게 입력한 문자수 세기
		for(i = 0; i < y_pos; i++) {
			len = strlen(article->page2[i]);
			for(j = 0; j < len; j++) {
				total++;
				if(input_buf[1][i][j] == article->page2[i][j])
					correct_count++;
			}
		}

		for(i = 0; i < x_pos; i++) {
			total++;
			if(input_buf[1][y_pos][i] == article->page2[y_pos][i])
				correct_count++;
		}
	}
	else {//첫 번째 페이지일 경우
```

```
		//첫 번째 페이지에서 올바르게 입력한 문자수 세기
		for(i = 0; i < y_pos; i++) {
			len = strlen(article->page1[i]);
			for(j = 0; j < len; j++) {
				total++;
				if(input_buf[0][i][j] == article->page1[i][j])
					correct_count++;
			}
		}

		for(i = 0; i < x_pos; i++) {
			total++;
			if(input_buf[0][y_pos][i] == article->page1[y_pos][i])
				correct_count++;
		}
	}

	//입력한 데이터가 없다면 정확도는 0
	if(total == 0)
		return 0;
	//정확도 계산
	result = correct_count * 100 / total;
	return result;
}

//긴 글 연습에서 현재 타수를 계산하는 함수
```

```
int cal_speed_for_long(char input_buf[][NUM_OF_ROWS][MAX_SIZE], Article *article, int page_num, int x_pos, int y_pos, struct timeval *start, struct timeval *end) {
	//시작 시간부터 입력받은 시간까지의 기간
	long long seconds = end->tv_sec - start->tv_sec;
	//현재 타수
	int speed;
	//반복문을 위해 사용
	int i, j;
	//정확히 입력한 문자 개수
	int correct_count = 0;
	//입력한 문자 개수
	int total = 0;
	//문장 길이
	int len;

	//입력을 시간한 상태이면 현재 타수는 0
	if(seconds == 0)
		return 0;

	//두 번째 페이지인 경우
	if(page_num == 1) {
		//첫 번째 페이지에서 올바르게 입력한 문자수 세기
		for(i = 0; i < 5; i++) {
			len = strlen(article->page1[i]);
			for(j = 0; j < len; j++) {
				total++;
```

```
                        if(input_buf[0][i][j] == article->page1[i][j])
                                correct_count++;
                }
        }

        //두 번째 페이지에서 올바르게 입력한 문자수 세기
        for(i = 0; i < y_pos; i++) {
                len = strlen(article->page2[i]);
                for(j = 0; j < len; j++) {
                        total++;
                        if(input_buf[1][i][j] == article->page2[i][j])
                                correct_count++;
                }
        }

        for(i = 0; i < x_pos; i++) {
                total++;
                if(input_buf[1][y_pos][i] == article->page2[y_pos][i])
                        correct_count++;
        }
}
else {//첫 번째 페이지인 경우
        //첫 번째 페이지에서 올바르게 입력한 문자수 세기
        for(i = 0; i < y_pos; i++) {
                len = strlen(article->page1[i]);
                for(j = 0; j < len; j++) {
```

```
                                        total++;
                                        if(input_buf[0][i][j] == article->page1[i][j])
                                                correct_count++;
                                }
                        }

                        for(i = 0; i < x_pos; i++) {
                                total++;
                                if(input_buf[0][y_pos][i] == article->page1[y_pos][i])
                                        correct_count++;
                        }
                }

                //현재 타수 계산
                speed = (correct_count * 60) / seconds;
                return speed;
}

//긴 글 연습을 실행시키는 함수
void exercise_long(void) {
                Article article[LONG_SIZE] = {
                        {//0
                                {
                                        "The Selfish Giant",
                                        "Every afternoon, as they were coming from school, the children
used",
```

```
                "to go and play in the Giant₩'s garden.",
                "It was a large lovely garden, with soft green grass. Here and there",
                "over the grass stook beautiful flowers like stars, and there were",
            },
            {
                "twelve peachtrees that in the springtime broke out into delicate blos-",
                "soms of pink and pearl, and in the autumn bore rich fruit. The birds",
                "sat in the trees and sang so sweetly that the children used to stop",
                "their games in order to listen to them. ₩"How happy we are here!₩" they",
                "cried to each other.",
            }
        },
        {//1
            {
                "The Elves and the Shoemaker",
                "There was once a shoemaker, who, through no fault of his own, became",
                "so poor that at last he had nothing left but just enough leather to",
                "make one pair of shoes. He cut out the shoes at night, so as to set to",
                "work upon them next morning; and as he had a good conscience, he laid",
```

```
                },
                {
                        "himself quietly down in his bed, committed himself to heaven,
and fell",
                        "asleep. In the morning, after he had said his prayers, and was
going",
                        "to get to work, he found the pair of shoes made and finished,
and stan-",
                        "ding on his table. He was very much astonished, and could not
tell",
                        "what to think, and he took the shoes in his hand to examine
then more",
                }
        },
        {//2
                {
                        "Rapunzel",
                        "There once lived a man and his wife, who had long wished for
a child,",
                        "but in vain. Now there was at the back of their house a little
window",
                        "which overlooked a beautiful garden full of the finest vegetables
and",
                        "flowers; but there was a high wall all round it, and no one
ventured",
                },
                {
                        "into it, for it belonged to a witch of great might, and of whom
all",
```

```
                "the world was afraid. One day, when the wife was standing at
the win-",
                "dow, and looking into the garden, she saw a bed filled with the
finest",
                "rampion; and it looked so fresh and green that she began to
wish for",
                "some; and at length she longed for it greatly.",
            }
        },
        {//3
            {
                "The Wind and the Sun",
                "The North Wind was rushing along and blowing the clouds as
he passed.",
                "₩"Who is so strong as I?₩" he cried.",
                "₩"I am even stronger than the sun.₩"",
                "₩"Can you show that you are stronger?₩" asked the Sun.",
            },
            {
                "₩"A traveler is coming over the hill,₩" said the Wind. ₩"Let us
see which",
                "of us can first make him take off his long cloak.",
                "The one who succeeds will prove himself the stronger.₩"",
                "The North Wind began first. He blew a gale, tore up trees, and
raised",
                "clouds of dust.",
            }
```

```
            }
};
//글의 행 별 문장 길이
size_t article_size[NUM_OF_PAGES][NUM_OF_ROWS];
//article의 인덱스
int no;
//정확도와 현재 타수
int accuracy = 0, speed = 0;
//입력 문자
int input;
//입력 글
char input_buf[NUM_OF_PAGES][NUM_OF_ROWS][MAX_SIZE];
//페이지 번호, 0이면 첫 번째 페이지, 1이면 두 번째 페이지
int page_num = 0;
//처음 입력 시간을 알기 위한 플래그
bool is_first = true;
//처음 시간과 끝 시간
struct timeval start, end;
//커서 좌표 위치
int x_pos = 0, y_pos = 0;
//반복문을 위해 사용
int i, j;

//랜덤 숫자의 seed를 현재 시간으로 설정
srand(time(NULL));
//article 인덱스 랜덤으로 받기
```

```
no = rand() % LONG_SIZE;

//행 별 문장 길이 저장
for(i = 0; i < NUM_OF_ROWS; i++) {
        article_size[0][i] = strlen(article[no].page1[i]);
        article_size[1][i] = strlen(article[no].page2[i]);
}

memset(input_buf, 0x00, sizeof(input_buf));
while(1) {
        start_msg(4);
        move_cursor(0, 2);
        printf("정확도 : %3d%%₩t현재타수 : %3d₩n", accuracy, speed);

        move_cursor(0, 5);
        for(i = 0; i < NUM_OF_ROWS; i++) {
                if(page_num == 0)
                        printf("%s₩n", article[no].page1[i]);
                else
                        printf("%s₩n", article[no].page2[i]);
        }

        move_cursor(0, 12);
        for(i = 0; i <= y_pos; i++)
                printf("%s₩n", input_buf[page_num][i]);
```

```
                move_cursor(x_pos, y_pos + 12);
                input = getch();
                //ESC 입력 시 메뉴 복귀
                if(input == ESC)
                        return;
                else if(input == DEL) {//백스페이스 입력 시
                        if(x_pos <= 0) {//행에 입력한 데이터가 없을 경우
                                continue;
                        }
                        else {
                                x_pos--;//인덱스 감소
                                input_buf[page_num][y_pos][x_pos] = 0;
                        }
                        //정확도 계산
                        accuracy = cal_accuracy_for_long(input_buf, &article[no], page_num,
x_pos, y_pos);
                        //현재 타수 계산
                        speed = cal_speed_for_long(input_buf, &article[no], page_num, x_pos,
y_pos, &start, &end);
                }
                else {
                        //처음 입력 시
                        if(is_first) {
                                //처음 시간 구하기
                                gettimeofday(&start, NULL);
                                is_first = false;
```

```
}
else//현재까지 시간 구하기
        gettimeofday(&end, NULL);


//엔터 입력 시
if(input == '₩n') {
        x_pos = 0;
        y_pos++;
}
if(y_pos == 5) {//한 페이지가 다 끝난 경우
        //첫 번째 페이지인 경우
        if(page_num == 0) {
                //두 번째 페이지로 이동
                page_num = 1;
                y_pos = 0;
        }
        else//두 번째 페이지인 경우
                break;
}


if(input != '₩n') {
        input_buf[page_num][y_pos][x_pos++] = input;
        input_buf[page_num][y_pos][x_pos] = 0;
        //정확도 계산
        accuracy = cal_accuracy_for_long(input_buf, &article[no],
page_num, x_pos, y_pos);
```

```
                                        //현재 타수 계산
                                        speed = cal_speed_for_long(input_buf, &article[no], page_num,
x_pos, y_pos, &start, &end);
                                }
                        }
                }

                //최종 결과
                start_msg(4);
                move_cursor(0, 2);
                printf("정확도 : %3d%%₩t현재타수 : %3d₩n", accuracy, speed);
                move_cursor(0, 5);
                printf("종료하려면 ESC를 누르세요.");
                move_cursor(0, 12);

                //ESC 입력 시 메뉴 복귀
                while(getch() != ESC);
}

//프로그램을 종료시키는 함수
void end(void) {
                clear();
                exit(0);
}

//오류를 알리는 함수
```

```
void error(void) {
        fprintf(stderr, "잘못된 번호를 입력하였습니다.₩n");
        sleep(1);
        clear();
}
```

2. 함수 상세 설명

1) void clear(void)

- 기능 : 화면을 clear하는 함수
- 반환값 : 없음
- 파라미터 : 없음

2) void move_cursor(int x, int y)

- 기능 : (x, y) 좌표로 화면애서 커서를 이동시키는 함수
- 반환값 : 없음
- 파라미터
  - x, y : 커서의 (x, y) 좌표

3) int getch(void)

- 기능 : 한 문자를 입력받는 함수, 입력한 문자가 화면에 보이지 않는다.
- 반환값 : 입력 받은 문자
- 파라미터 : 없음

4) void start_msg(int no)

- 기능 : 시작 메시지를 출력하는 함수
- 반환값 : 없음
- 파라미터 : 프로그램 기능 번호

5) int menu(void)

- 기능 : 메뉴를 출력하고 메뉴 번호를 입력받는 함수
- 반환값 : 입력 받은 프로그램 기능 번호
- 파라미터 : 없음

6) int cal_accuracy_for_short(char *input, char *sen, int idx)

- 기능 : 짧은 글 연습에서 정확도를 계산하는 함수
- 반환값 : 정확도
- 파라미터
  - input : 입력 받은 문장
  - sen : 입력해야 하는 문장
  - index : 현재까지 입력한 문장의 길이

7) int cal_speed_for_short(char *input, char *sen, int idx, struct timeval *start, struct timeval *end)

- 기능 : 짧은 글 연습에서 현재 타수를 계산하는 함수
- 반환값 : 현재 타수
- 파라미터
  - input : 입력 받은 문장
  - sen : 입력해야 하는 문장
  - index : 현재까지 입력한 문장의 길이
  - start : 시작 시간
  - end : 마지막 시간

8) int cal_accuracy_for_long(char input_buf[][NUM_OF_ROWS][MAX_SIZE], Article *article, int page_num, int x_pos, int y_pos)

- 기능 : 긴 글 연습에서 정확도를 계산하는 함수
- 반환값 : 정확도
- 파라미터
  - input : 입력 받은 글
  - article : 입력해야 하는 글
  - page_num : 현재 페이지 번호
  - x_pos, y_pos : 글 내에서의 커서 위치

9) int cal_speed_for_long(char input_buf[][NUM_OF_ROWS][MAX_SIZE], Article *article, int page_num, int x_pos, int y_pos, struct timeval *start, struct timeval *end)

- 기능 : 긴 글 연습에서 현재 타수를 계산하는 함수
- 반환값 : 현재 타수
- 파라미터
  - input : 입력 받은 글
  - article : 입력해야 하는 글
  - page_num : 현재 페이지 번호
  - x_pos, y_pos : 글 내에서의 커서 위치
  - start : 시작 시간
  - end : 마지막 시간

10) void exercise_pos(void)

- 기능 : 자리 연습을 실행시키는 함수
- 반환값 : 없음
- 파라미터 : 없음

11) void exercise_word(void)

- 기능 : 낱말 연습을 실행시키는 함수
- 반환값 : 없음
- 파라미터 : 없음

12) void exercise_short(void)

- 기능 : 짧은 글 연습을 실행시키는 함수
- 반환값 : 없음
- 파라미터 : 없음

13) void exercise_long(void)

- 기능 : 긴 글 연습을 실행시키는 함수
- 반환값 : 없음
- 파라미터 : 없음

14) void end(void)

- 기능 : 프로그램을 종료시키는 함수
- 반환값 : 없음
- 파라미터 : 없음

15) void error(void)

- 기능 : 오류를 알리는 함수
- 반환값 : 없음
- 파라미터 : 없음

3. 역할 분담

- 홍상원

    - 짧은 글 연습, 긴 글 연습 구현
    - 주석 작업
    - 문서화 작업
    - 리펙토링
- 김연수
    - 자리 연습, 낱말 연습 구현
    - 주석 작업
    - 문서화 작업
    - 평가 항목 구현 여부 테스트
- 김희원
    - 자리 연습, 낱말 연습 구현
    - 주석 작업
    - 문서화 작업
    - 평가 항목 구현 여부 테스트