

2주차 (반복문, 조건문)

1. 조건문 (Conditional Statements)

1.1 `if` 문

[문법](#)

[예제](#)

1.2 `if-else` 문

[문법](#)

[예제](#)

1.3 `else if` 문

[문법](#)

[예제](#)

[추가 팁](#)

1.4 `switch` 문

[문법](#)

[예제](#)

[주의점](#)

1.5 중첩 조건문

[예제](#)

2. 반복문(Loops)

2.1 `for` 문

[문법](#)

[예제](#)

[디테일](#)

2.2 `while` 문

[문법](#)

[예제](#)

[주의점](#)

2.3 `do-while` 문

[문법](#)

[예제](#)

2.4 반복문 제어: `break` 와 `continue`

[예제 1](#)

[예제 2](#)

1. 조건문 (Conditional Statements)

조건문은 프로그램이 특정 조건에 따라 다른 동작을 수행하도록 만드는 도구입니다. 마치 일상에서 "만약 날씨가 맑으면 산책을 하고, 비가 오면 집에 머문다"처럼 조건에 따라 행동을 결정하는 것과 같습니다.

1.1 `if` 문

가장 기본적인 조건문으로, 조건이 참일 때만 특정 코드를 실행합니다.

문법

```
if (조건) {  
    // 참일 경우에 실행되는 코드 블록  
}
```

- **조건:** 참(**true**) 또는 거짓(**false**)으로 평가되는 식 (예: `x > 5`)
- **중괄호 {}**는 실행할 코드가 여러 줄일 때 필수이며, 한 줄일 경우 생략 가능합니다. (권장하지 않음).

예제

나이 제한 확인

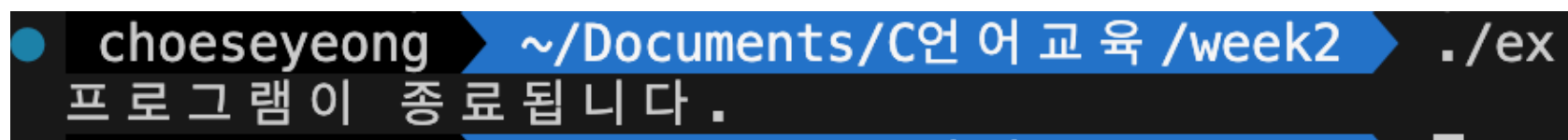
```
#include <stdio.h>

int main() {
    int age = 15;

    if (age >= 19) {
        printf("영화 관람이 가능합니다.\n");
    }

    printf("프로그램이 종료됩니다.\n");
    return 0;
}
```

- 실행 결과



- 설명: `age` 가 15이므로 `age >= 19` 는 거짓이 되어 `if` 안의 `printf` 가 실행되지 않습니다.

1.2 if-else 문

조건이 참일 때와 거짓일 때 각각 다른 작업을 수행합니다.

문법

```
if (조건) {
    // 조건이 참일 때 실행
} else {
    // 조건이 거짓일 때 실행
}
```

예제

시험 합격 여부

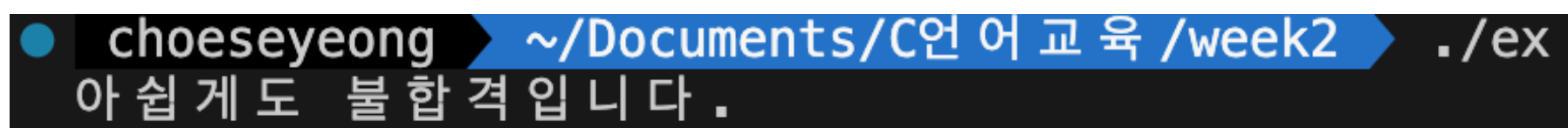
```
#include <stdio.h>

int main() {
    int score = 55;

    if (score >= 60) {
        printf("합격을 축하합니다!\n");
    } else {
        printf("아쉽게도 불합격입니다.\n");
    }

    return 0;
}
```

- 실행 결과



- 설명 : `score` 가 55로 60 미만이므로, `else` 블록이 실행됩니다.

1.3 else if 문

여러 조건을 순차적으로 확인할 때 사용합니다.

문법

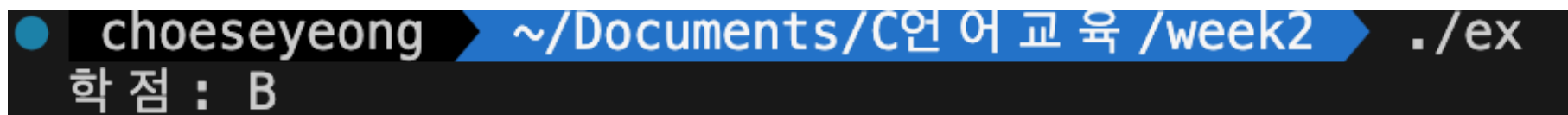
```
if (조건1) {  
    // 조건1이 참일 때  
} else if (조건2) {  
    // 조건1이 거짓이고 조건2가 참일 때  
} else {  
    // 모든 조건이 거짓일 때  
}
```

예제

점수에 따른 학점

```
#include <stdio.h>  
  
int main() {  
    int score = 87;  
  
    if (score >= 90) {  
        printf("학점: A\n");  
    } else if (score >= 80) {  
        printf("학점: B\n");  
    } else if (score >= 70) {  
        printf("학점: C\n");  
    } else if (score >= 60) {  
        printf("학점: D\n");  
    } else {  
        printf("학점: F\n");  
    }  
  
    return 0;  
}
```

- 출력



```
choeseyeong ~/Documents/C언어교육/week2 ./ex  
학 점 : B
```

- 설명: `score`가 87이므로 첫 번째 조건(`>= 90`)은 거짓, 두 번째 조건(`>= 80`)은 참이 되어 "학점: B"가 출력됩니다.

추가 팁

- 조건은 위에서 아래로 순차적으로 평가되며, 참인 첫 번째 조건을 만나면 나머지는 무시됩니다.
- 비교 연산자: `>`(초과), `<`(미만), `>=`(이상), `<=`(이하), `==`(같음), `!=`(다름)
- 논리 연산자: `&&`(AND), `||`(OR), `!`(NOT)
 - 예: `if (score >= 80 && score < 90) → "80 이상 90 미만"`

1.4 switch 문

특정 변수의 값에 따라 여러 경우 중 하나를 선택할 때 사용합니다. `if-else` 의 대안으로, 코드가 깔끔해지는 경우가 많습니다.

문법

```
switch (식) {
    case 값1:
        // 값1일 때 실행
        break;
    case 값2:
        // 값2일 때 실행
        break;
    default:
        // 모든 case에 해당하지 않을 때 실행
}
```

- **식**: 정수나 문자처럼 단일 값을 가지는 표현식 (예: `int`, `char`)
- **case**: 식의 값과 비교할 상수
- **break**: 해당 `case` 실행 후 `switch` 를 빠져나감 (없으면 다음 `case` 까지 실행됨)
- **default**: 모든 `case` 에 해당하지 않을 때 실행 (선택 사항)

예제

메뉴 선택

```
#include <stdio.h>

int main() {
    int choice;

    printf("메뉴를 선택하세요 (1-3): ");
    scanf("%d", &choice);

    switch (choice) {
        case 1:
            printf("햄버거를 선택했습니다.\n");
            break;
        case 2:
            printf("피자를 선택했습니다.\n");
            break;
        case 3:
            printf("샐러드를 선택했습니다.\n");
            break;
        default:
            printf("잘못된 선택입니다.\n");
    }

    return 0;
}
```

- 출력

```

● choeseyeong ~/Documents/C언어교육/week2 ./ex
메뉴를 선택하세요 (1-3): 1
햄버거를 선택했습니다.

● choeseyeong ~/Documents/C언어교육/week2 ./ex
메뉴를 선택하세요 (1-3): 2
피자를 선택했습니다.

● choeseyeong ~/Documents/C언어교육/week2 ./ex
메뉴를 선택하세요 (1-3): 3
샐러드를 선택했습니다.

● choeseyeong ~/Documents/C언어교육/week2 ./ex
메뉴를 선택하세요 (1-3): 4
잘못된 선택입니다.

```

- 설명: 사용자가 입력한 값에 따라 해당 메뉴가 출력됩니다.

주의점

- `break` 를 잊으면 다음 `case` 까지 "떨어져 내려가"(fall-through) 실행됩니다.
 - 예: `break` 없이 `case 1` 실행 시 `case 2` 까지 이어짐.
- `switch` 는 실수(`float`, `double`)나 문자열(`string`)을 직접 사용할 수 없습니다.

1.5 중첩 조건문

`if` 문 안에 또 다른 `if` 문을 넣어 복잡한 조건을 처리할 수 있습니다.

예제

나이와 키 제한

```

#include <stdio.h>

int main() {
    int age = 12;
    int height = 150;

    if (age >= 10) {
        if (height >= 140) {
            printf("놀이기구를 탈 수 있습니다.\n");
        } else {
            printf("키가 부족합니다.\n");
        }
    } else {
        printf("나이가 너무 어립니다.\n");
    }

    return 0;
}

```

- 출력

```

● choeseyeong ~/Documents/C언어교육/week2 ./ex
놀이기구를 탈 수 있습니다.

```

- **설명:** `age >= 10` 이 참이고, 중첩된 `height >= 140` 도 참이므로 놀이기구 탑승이 가능합니다.

2. 반복문(Loops)

반복문은 동일한 작업을 여러 번 실행할 때 사용합니다. 예를 들어, "책을 5번 읽어"처럼 반복 횟수나 조건을 지정할 수 있습니다.

2.1 for 문

정확한 반복 횟수를 알고 있을 때 사용합니다.

문법

```
for (초기화; 조건; 증감식) {  
    // 반복할 코드  
}
```

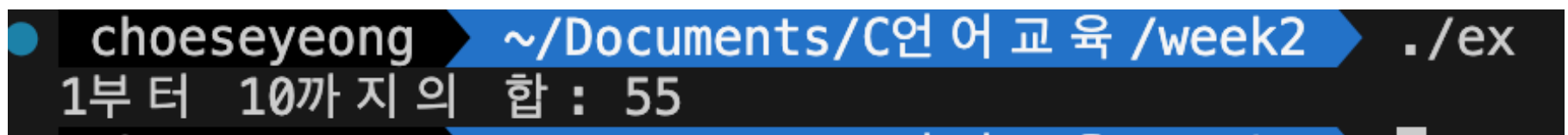
- **초기화:** 반복 시작 전 변수 설정 (예: `int i = 0`)
- **조건:** 참일 동안 반복 (예: `i < 10`)
- **증감식:** 반복 후 변수 업데이트 (예: `i++`)

예제

: 1부터 10까지의 합

```
#include <stdio.h>  
  
int main() {  
    int sum = 0;  
  
    for (int i = 1; i <= 10; i++) {  
        sum = sum + i;  
    }  
  
    printf("1부터 10까지의 합: %d\n", sum);  
    return 0;  
}
```

- 출력



```
choeseyeong ~/Documents/C언어교육/week2 ./ex  
1부터 10까지의 합 : 55
```

- **설명:** `i` 가 1부터 10까지 증가하며 `sum` 에 더해집니다.

디테일

- `i++` 는 `i = i + 1` 과 같습니다.
- 조건이 처음부터 거짓이면 반복문은 실행되지 않습니다.
 - ex) `for(int i = 11; i <= 10; i++)`

2.2 while 문

조건이 참인 동안 반복하며, 횟수를 미리 모를 때 유용합니다.

문법

```
while (조건) {  
    // 조건이 참일 때 반복  
}
```

예제

사용자가 원하는 합계에 도달할 때까지 숫자 입력받기

```
#include <stdio.h>  
  
int main() {  
    int sum = 0;        // 현재 합계를 저장할 변수  
    int input;          // 사용자가 입력한 숫자  
    int target = 100;    // 목표 합계 (사용자가 바꿀 수 있음)  
    int attempts = 0;    // 입력 횟수 카운트  
  
    printf("목표 합계는 %d입니다. 숫자를 입력하세요.\n", target);  
  
    // 합계가 목표에 도달할 때까지 반복  
    while (sum < target) {  
        printf("현재 합계: %d | 숫자를 입력하세요: ", sum);  
        scanf("%d", &input); // 사용자 입력 받기  
  
        // 음수 입력 방지  
        if (input < 0) {  
            printf("음수는 허용되지 않습니다. 다시 입력하세요.\n");  
            continue; // 이번 반복 건너뛰고 다시 입력 받음  
        }  
  
        sum += input;    // 입력값을 합계에 추가  
        attempts++;      // 시도 횟수 증가  
  
        // 합계가 목표를 초과하면 경고  
        if (sum > target) {  
            printf("목표를 초과했습니다! (현재 합계: %d)\n", sum);  
        }  
    }  
  
    // 결과 출력  
    printf("목표 %d에 도달했습니다!\n", target);  
    printf("최종 합계: %d, 입력 횟수: %d\n", sum, attempts);  
  
    return 0;  
}
```

- 출력

```
● choeseyeong ~/Documents/C언어교육/week2 ./ex
목표 합계는 100입니다. 숫자를 입력하세요.
현재 합계 : 0 | 숫자를 입력하세요 : 30
현재 합계 : 30 | 숫자를 입력하세요 : 50
현재 합계 : 80 | 숫자를 입력하세요 : -10
음수는 허용되지 않습니다. 다시 입력하세요.
현재 합계 : 80 | 숫자를 입력하세요 : 25
목표를 초과했습니다! (현재 합계 : 105)
목표 100에 도달했습니다!
최종 합계 : 105, 입력 횟수 : 3
```

코드 분석

1. 초기 설정 :

- `sum = 0` : 합계를 0부터 시작.
- `target = 100` : 목표 합계로, 나중에 변수로 바꿀 수도 있음.
- `attempts = 0` : 입력 횟수를 추적.

2. while 루프:

- `while (sum < target)` : 합계가 목표 미만일 때만 반복.
- 사용자가 입력한 값(`input`)을 받아 `sum`에 더하며, 조건이 충족될 때까지 계속.

3. 입력 검증:

- `if (input < 0)` : 음수를 입력하면 경고 후 `continue`로 다시 입력 요구.

4. 종료 조건:

- `sum >= target`이 되면 루프 종료.

5. 결과 출력:

주의점

- 조건을 잘못 설정하면 “무한 루프”에 빠질 수 있습니다. (ex. `while(1)`)

2.3 do-while 문

코드를 먼저 실행한 뒤 조건을 확인합니다. 최소 한 번은 실행 보장됩니다.

문법

```
do {
    // 반복할 코드
} while (조건);
```

- 세미콜론(;)이 끝에 붙는 점에 주의하세요.

예제

양수 입력 강제

```
#include <stdio.h>

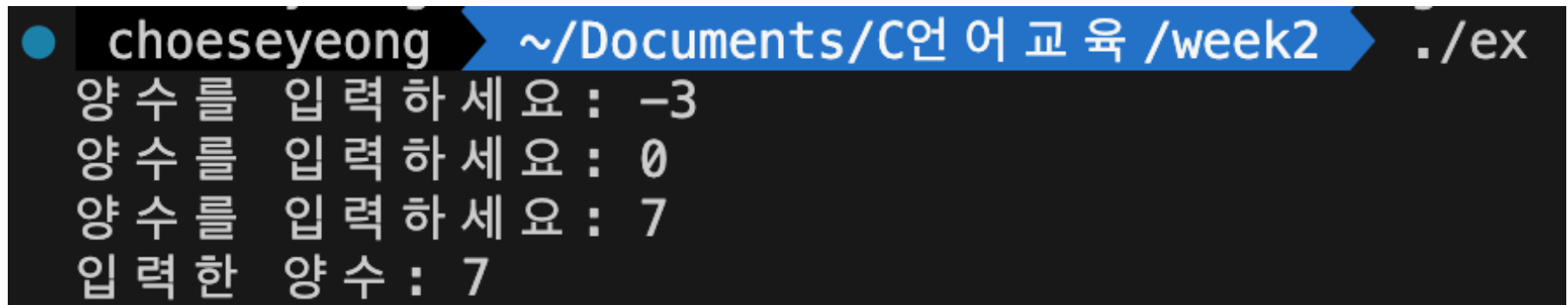
int main() {
    int num;
```



```
do {
    printf("양수를 입력하세요: ");
    scanf("%d", &num);
} while (num <= 0);

printf("입력한 양수: %d\n", num);
return 0;
}
```

- 출력



```
choeseyeong ~/Documents/C언어교육/week2 ./ex
양수를 입력하세요: -3
양수를 입력하세요: 0
양수를 입력하세요: 7
입력한 양수: 7
```

- 설명: `num <= 0` 이 거짓이 될 때까지 반복합니다.

2.4 반복문 제어: `break` 와 `continue`

- `break`: 반복문을 즉시 종료.
- `continue`: 현재 반복을 건너뛰고 다음 반복으로 이동.

예제 1

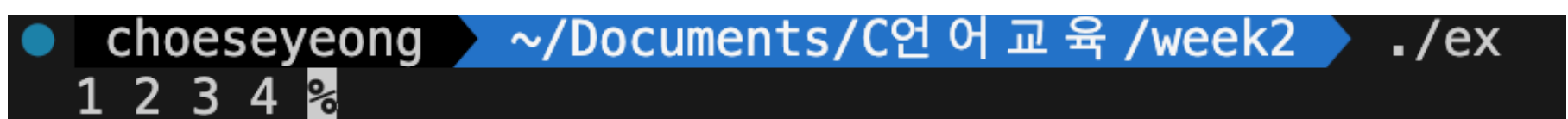
5에서 멈추기

```
#include <stdio.h>

int main() {
    for (int i = 1; i <= 10; i++) {
        if (i == 5) {
            break; // i가 5일 때 반복 종료
        }
        printf("%d ", i);
    }

    return 0;
}
```

- 출력



```
choeseyeong ~/Documents/C언어교육/week2 ./ex
1 2 3 4 %
```

예제 2

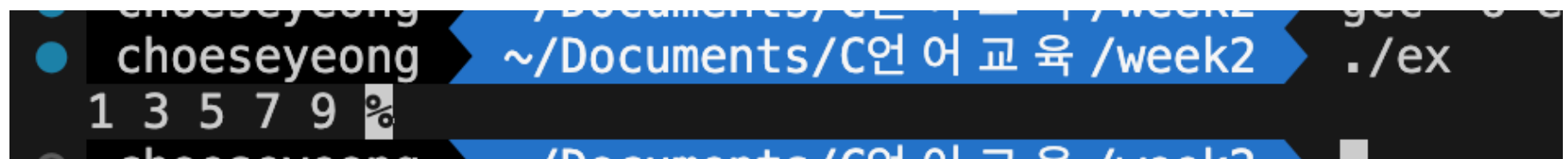
짝수 건너뛰기

```
#include <stdio.h>

int main() {
    for (int i = 1; i <= 10; i++) {
        if (i % 2 == 0) {
            continue; // 짝수일 때 건너뛴
        }
        printf("%d ", i);
    }

    return 0;
}
```

- 출력



```
choeseyeong ~/Documents/C언어교육/week2
choeseyeong ~/Documents/C언어교육/week2 ./ex
1 3 5 7 9 %
```