출처: https://inf.run/zSrvA ↔ (https://www.youtube.com/@weekendcode)

반복문이란?

특정 코드를 계속 반복하는 구문으로, for / while / do..while 형태로 시험 출제

반복문의 기본 형태	설명과 실행순서

for (초기식; 조건식; 증감식) {
  // 반복 실행할 코드
}
	초기식: 반복문이 처음 시작할 때 단 한 번 실행됩니다.
	조건식: 조건이 참(true)인 동안에 반복문을 실행합니다.
	조건식이 거짓(false)이 되면 반복문이 종료됩니다.
	증감식: 반복문이 한 번 실행될 때마다 조건식을 검사하기 전에 실행됩니다.

(※ 실행순서 중요: 1. 초기식 / 2. 조건식 / 3. 코드 실행 / 4. 증감식 / 5. 조건식..)	

실제 예시를 살펴볼까요?

예시1

```
#include <stdio.h>
int main() {
    for (int i = 0; i < 5; i++) {
        printf("i의 값: %d\n", i);
    }
    return 0;
}
```

출력값:

```
i의 값: 0
i의 값: 1
i의 값: 2
i의 값: 3
i의 값: 4
```

예시2

```
int main() {
    for (int i = 0; i < 3; i++) {
        printf("i의 값: %d\n", i);
  }
}
```

출력값:

```
i의 값: 0
i의 값: 1
i의 값: 2
```

수업 영상을 보고 실행순서를 직접 작성해보세요:

중첩 반복문 (이중 for문)

```
#include <stdio.h>

int main() {
    for (int i = 2; i <= 9; i++) {
        for (int j = 1; j <= 9; j++) {
            if(i%2 == 0)
            printf("%d*%d = %d ", i, j, i * j);
        }
        printf("\n");
    }
    return 0;
}
```

출력값:

```
2*1 = 2 2*2 = 4 2*3 = 6 2*4 = 8 2*5 = 10 2*6 = 12 2*7 = 14 2*8 = 16 2*9 = 18
4*1 = 4 4*2 = 8 4*3 = 12 4*4 = 16 4*5 = 20 4*6 = 24 4*7 = 28 4*8 = 32 4*9 = 36
6*1 = 6 6*2 = 12 6*3 = 18 6*4 = 24 6*5 = 30 6*6 = 36 6*7 = 42 6*8 = 48 6*9 = 54
8*1 = 8 8*2 = 16 8*3 = 24 8*4 = 32 8*5 = 40 8*6 = 48 8*7 = 56 8*8 = 64 8*9 = 72
```

생각해보기: 첫 줄에서, 왜 앞이 2이고, 뒤의 숫자가 변할까?

< 중첩 반복문 해석하기 >

```
int main() {
    for (int outer = 2; outer <= 5; outer++) {
        for (int inner = 3; inner <= 4; inner++) {
            printf("%d*%d=%d ", outer, inner, outer * inner);
        }
        printf("\n");
    }
    return 0;
}
```

출력문:

```
2*3=6 2*4=8
3*3=9 3*4=12
4*3=12 4*4=16
5*3=15 5*4=20
```

코드	출력값

#include <stdio.h>
int main() {
  int i = 0;
  for (; i < 5; i++) {
  printf("i의 값: %d\n", i);
  }
}
	i의 값: 0
	i의 값: 1
	i의 값: 2
	i의 값: 3
	i의 값: 4

#include <stdio.h>
int main() {
  int i = 1;
  for (; i <= 10; i++) {
  printf("%d * %d = %d\n", 2, i, 2 * i);
  }
}
	2 * 1 = 2
	2 * 2 = 4
	2 * 3 = 6
	2 * 4 = 8
	2 * 5 = 10
	2 * 6 = 12
	2 * 7 = 14
	2 * 8 = 16
	2 * 9 = 18
	2 * 10 = 20

#include <stdio.h>
int main() {
  for (int i = 0; ; i++) {
  if (i >= 5) {
  break;
  }
  printf("i의 값: %d\n", i);
  }
}
	i의 값: 0
	i의 값: 1
	i의 값: 2
	i의 값: 3
	i의 값: 4

#include <stdio.h>
int main() {
  for (int i = 1; ; i++) {
  printf("%d * %d = %d\n", 2, i, 2 * i);
  if (i >= 10) {
  break;
  }
  }
}
	2 * 1 = 2
	2 * 2 = 4
	2 * 3 = 6
	2 * 4 = 8
	2 * 5 = 10
	2 * 6 = 12
	2 * 7 = 14
	2 * 8 = 16
	2 * 9 = 18
	2 * 10 = 20

코드	출력값

#include <stdio.h>
int main() {
  for (int i = 0; i < 5;) {
  printf("i의 값: %d\n", i);
  i++;
  }
}
	i의 값: 0
	i의 값: 1
	i의 값: 2
	i의 값: 3
	i의 값: 4

#include <stdio.h>
int main() {
  for (int i = 1; i <= 10;) {
  printf("%d * %d = %d\n", 2, i, 2 * i);
  i++;
  }
}
	2 * 1 = 2
	2 * 2 = 4
	2 * 3 = 6
	2 * 4 = 8
	2 * 5 = 10
	2 * 6 = 12
	2 * 7 = 14
	2 * 8 = 16
	2 * 9 = 18
	2 * 10 = 20

반복문 – while

반복문의 기본 형태	설명과 실행순서

while (조건식) {
  // 반복 실행할 코드
}
	while은 조건이 참인 동안 계속 반복하는 형태의 반복문
	조건을 달성해서 빠져나가거나, 중간에 break라는 키워드로 문장을 빠져나가야함

(※ 실행순서: 조건식을 판단한 이후에 아래 블록 수행, 수행 후 다시 조건식 확인..)	

코드	출력값

#include <stdio.h>
 	
int main() {
  int i = 0; // 초기 변수 설정
  while (i < 5) {
  printf("i의 값: %d\n", i);
  i++;
  }
  return 0;
}
	i의 값: 0
	i의 값: 1
	i의 값: 2
	i의 값: 3
	i의 값: 4

#include <stdio.h>
 	
int main() {
  int i = 0;
  while (1) {
  printf("i의 값: %d\n", i);
  if (i >= 4) {
  break;
  }
  i++;
  }
  return 0;
}
	i의 값: 0
	i의 값: 1
	i의 값: 2
	i의 값: 3
	i의 값: 4

반복문 - for, while 섞은 예제

코드	출력값

#include <stdio.h>
 	
int main() {
for (int i = 1; i <= 4; i++) {
printf("현재 숫자: %d\n", i);
int check = i;
 	
while (check % 2 == 0) {
printf("%d는 짝수입니다.\n", check);
break;
}
}
return 0;
}
	현재 숫자: 1
	현재 숫자: 2
	2는 짝수입니다.
	현재 숫자: 3
	현재 숫자: 4
	4는 짝수입니다.

반복문 – do..while

반복문의 기본 형태	설명과 실행순서

do {
  // 반복 실행할 코드
} while (조건식);
	do..while 반복문은 조건을 검사하기 전에 코드를 먼저 실행합니다.
	반복 블록을 최소 한 번 실행한 후 조건을 검사하여 반복을 계속할지 결정합니다.

(※ 실행순서: 일단 do 블록 안에 있는 것을 최초로 실행하고, 조건을 판단하여 참인 경우 블록 재수행)	

코드	출력값

int main() {
  int i = 1;
  do {
  printf("i의 값: %d\n", i);
  i++; // 변수 값 증가
  } while (i <= 5);
}
	출력값:
	i의 값: 1
	i의 값: 2
	i의 값: 3
	i의 값: 4
	i의 값: 5

while과 do..while의 차이를 명확하게 보자. (전처리기 생략)

코드	출력값

int main() {
  int i = 10; // 초기 변수 설정
 	
  do {
  printf("do..while: i의 값: %d\n", i);
  i++;
  } while (i < 10);
  return 0;
}
	do..while: i의 값: 10

int main() {
  int i = 10; // 초기 변수 설정
 	
  while (i < 10) {
  printf("while: i의 값: %d\n", i);
  i++;
  }
  return 0;
}
	(아무 것도 출력되지 않음)

반복문 예시 모음

코드	출력값

#include <stdio.h>
 	
int main() {
 	
  int count = 0;
  int num = 1;
 	
  while (count < 7) {
  if (num % 3 == 0 || num % 5 == 0) {
  printf("%d_", num);
  count++;
  }
  num++;
  }
  printf("\n");
 	
  return 0;
}
	3_5_6_9_10_12_15_

#include <stdio.h>
 	
int main() {
 	
  for (int i = 6; i <= 8; i++) {
  for (int j = 1; j <= 3; j++) {
  printf("%d*%d=%d ", i, j, i*j);
  }
  printf("\n");
  }
 	
  return 0;
}
	6*1=6 6*2=12 6*3=18
	7*1=7 7*2=14 7*3=21
	8*1=8 8*2=16 8*3=24

코드	출력값

#include <stdio.h>
 	
int main() {
 	
int i = 1;
int sum = 0;
 	
for (; i <= 3; i++) {
int j = 1;
for (; j <= 3;) {
printf("%d*%d=%d//", i, j, i*j);
j++;
}
printf("\n");
}
 	
return 0;
}
	1*1=1//1*2=2//1*3=3//
	2*1=2//2*2=4//2*3=6//
	3*1=3//3*2=6//3*3=9//

#include <stdio.h>
 	
int main() {
int x = 10;
 	
for (int i = 5; i <= 0; i++) {
if (i % 2 == 0) {
printf("i = %d\n", i);
}
}
 	
printf("x = %d\n", x);
 	
return 0;
}
	x = 10