C언어 스터디 3차시

컴퓨터학부 23 유상원

목차

- 제어 흐름
- 함수
- 표준 라이브러리

제어 흐름

조건문

- 조건
  - 조건의 참과 거짓에 따라 코드 실행
  - 주로 비교 연산의 결과가 사용됨
  - C언어에서 0은 거짓, 0 이외의 값은 참으로 처리
  - 참과 거짓을 나타내기 위한 자료형으로 _Bool(C99)과 bool(C++)이 있음
- 조건문
  - 조건이 참일 때 특정한 코드 실행
- e.g. if, switch

if

- 형태: if(조건) ... [else ...]
- 조건이 참일 때, if문 뒤에 있는 코드를 실행
- 조건이 거짓일 때, else문 뒤에 있는 중괄호에 있는 코드를 실행
- else문은 선택 사항임

if

- e.g.

```
if(a == 1) {
    printf("a == 1");
} else {
    printf("a != 1");
}
```

if

- else문 뒤에 if문을 합쳐 else-if 문을 사용할 수 있음
- 형태: if(cond1) { } else if(cond2) { } else if(cond3) { } ... else { }
  → [if() {} else [if {} else [if {} else {}]]] 로 묶어서 볼 수 있음

if

- e.g.

```
if(n < 0) {
    printf("negative");
} else if(n > 0) {
    printf("positive");
} else {
    printf("zero");
}
```

switch

- 형태: switch(표현식) { case 값: ...; default: ...; }
- 표현식과 값을 비교하고, 일치하는 case 위치로 이동하여 코드 실행
- 동일한 값이 하나도 없는 경우 default 위치로 이동하여 코드 실행
- switch 문을 벗어나려면 break 사용
- default 문은 선택사항

switch

- e.g.

```
switch(n % 10) {
    case 1:
        printf("%dst\n", n);
    case 2:
        printf("%dnd\n", n);
    case 3:
        printf("%drd\n", n);
    default:
        printf("%dth\n", n);
}
```

switch

- e.g.

```
switch(n % 10) {
    case 1:
        printf("%dst\n", n);
        break;
    case 2:
        printf("%dnd\n", n);
        break;
    case 3:
        printf("%drd\n", n);
        break;
    default:
        printf("%dth\n", n);
        break;
}
```

연습

- 1330 두 수 비교하기

반복문

- 특정 조건이 만족되는 동안 반복
- e.g. while, do-while, for

while

- 형태: while(조건) ...
- 조건이 만족되는 동안 반복
- 조건을 잘 설정해 무한 루프에 빠지지 않도록 조심하자

while

- e.g. “Hello” 10 번 출력

```
int t = 10;

while(t--) {
    printf("Hello ");
}
```

do-while

- 형태: do { } while(조건);
- 코드를 먼저 실행하고, 조건은 나중에 확인

do-while

- e.g. 10부터 1까지 카운트다운

```
int t = 10;
do {
    printf("%d\n", t);
    t --;
} while(t > 0);
```

do-while

- e.g. 10부터 1까지 카운트다운

```
int t = 10;
do {
    printf("%d\n", t);
} while(--t);
```

for

- 형태: for(초기식; 조건식; 증감식) ...

- while문이 조건을 만족 하는 동안 반복하는데 특화되었다면
- for문은 a부터 b까지 반복하는데 특화된 느낌?

for

- 형태: for(초기식; 조건식; 증감식) ...

- 조건식에는 while 문과 같이 반복 조건이 들어감
- 초기식에는 조건식에서 사용할 변수를 선언/초기화 하는 코드가 들어감
- 증감식에는 조건식에서 사용되는 변수를 증감하는 코드가 들어감
- 초기식, 조건식, 증감식은 선택 사항

for

- 형태: for(초기식; 조건식; 증감식) ...

- 초기식 → 조건식 확인 → 코드 실행 → 증감식 → 조건식 확인 → 코드 실행 → 증감식 → 조건식 확인 → ... → 증감식 → 조건식 확인 → 루프 종료

for

- e.g.

```
for(int i=0; i<10; i++) {
    printf("loop #%d\n", i);
}
```

- 참고: 초기식에서 변수를 선언하는 것은 C99 이후부터 가능

for

- e.g.

```
int i;
for(i=10; i>=1; i--) {
    printf("%d\n", i);
}
```

for

- e.g.

```
char *obfuscated = "Gdkkn";
for(int i=0; obfuscated[i] != '\0'; i++) {
    printf("%c", obfuscated[i]+1);
}
```

for

- e.g.

```
int i=10;
for(; i>=1;) {
    printf("%d\n", i);
    i--;
}
```

for

- e.g.

```
for(;;) {
    printf("endless loop\n");
}
```

break

- 반복문 탈출
- e.g.

```
for(;;) {
    printf("endless loop?\n");
    break;
}

printf("false\n");
```

continue

- 현재 루프를 멈추고, 다음 루프를 시작
- e.g.

```
for(int i=1; i<=10; i++) {
    if(i % 2) continue;
    printf("%d\n", i);
}
```

레이블과 goto

- 레이블의 위치로 실행 흐름을 변경
- 부작용이 많아 사용하지 않는게 바람직함

레이블과 goto

- e.g.

```
int main(void) {
    for(int i=0; i<10; i++) {
        if(i == 4) goto HATE;
        printf("%d\n", i);
    }

    return 0;

    HATE:
    printf("I hate 4 :(");
    return 0;
}
```

연습

- 2438 별 찍기 - 1

함수

함수

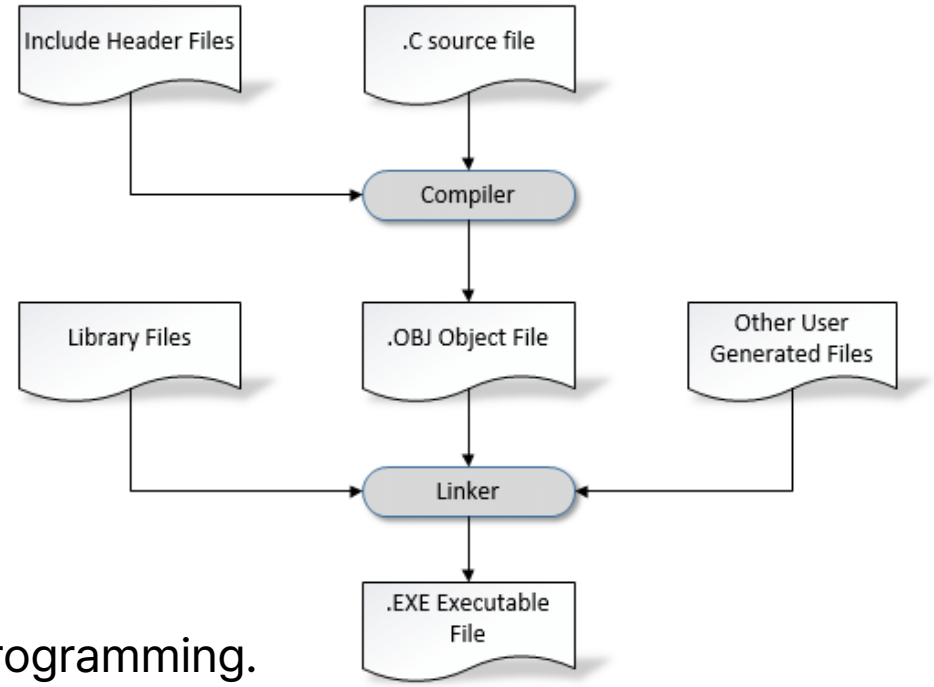
- 지금까지 남이 만든 함수를 호출했음 (e.g. printf, scanf)
- 이제는 직접 함수를 만들어볼 차례

함수

- 선언 (declaration): '이런 함수가 있어요'
  - e.g. `int sum(int a, int b);`
- 정의 (definition): '그 함수의 내용은 이래요'
  - e.g. `int sum(int a, int b) { return a + b; }`
- 정의만 해도 될 것 같은데, 굳이 선언도 해야 하나요?

함수

- 컴파일러는 각 소스코드 파일을 해석하여 목적파일을 생성
- 링커는 각 목적파일을 합쳐서 실행파일을 생성



*이미지 출처: Dawadi, Babu & Bhatta, Ram. (2015). Capsules of C Programming.

함수

- 다른 파일에 있는 함수를 끌어 쓰려면 어떡해야 할까?
  - 컴파일러와 링커가 다른 파일에 있는 함수를 복사/붙여넣기 하면 비효율적
  - 다른 파일에 있는 함수를 잠깐 실행하고 다시 돌아오면 되지 않을까?
  - 함수의 인자와 반환 값의 자료형을 알아야 한다.

- 다른 파일에 있는 함수의 인자와 반환 값을 어떻게 알아낼까?
  - 함수를 잠깐 실행하고 다시 돌아오도록 하는 것은 컴파일러와 링커의 몫
  - 헤더 파일에 함수의 선언을 작성하고, 그 헤더 파일을 불러와 사용
  - 우리가 `#include <stdio.h>`를 사용한 이유!

함수 선언

- 형태: [자료형] [함수이름]([매개변수 자료형] [매개변수 이름], ...);
- 매개변수는 없을 수도 있고, 여러 개가 있을 수도 있다.

- e.g. `int sum(int a, int b);`

함수 정의

- 형태: [자료형] [함수이름]([매개변수 자료형] [매개변수 이름], ...) { }
- 선언과 정의에서의 함수 이름, 자료형, 매개변수는 같아야 함
- void형이 아닌 이상 return으로 값을 반환해야 함

- e.g.

```
int sum(int a, int b) {
    return a + b;
}
```

연습

- op.c, op.h 파일 생성
- op.h에 int sum(int a, int b)와 int mul(int a, int b) 선언
- op.c에 sum, mul 정의
  - sum 함수에서는 return문과 + 연산자만 사용할 것
  - mul 함수에는 for문과 sum 함수만 사용할 것
- main.c에서 mul(a, b); 호출

표준 라이브러리

라이브러리

- 라이브러리: 쉽게 가져다 쓸 수 있도록 기능을 미리 구현해 놓은 것
- e.g. Win32API, POSIX, OpenGL, Vulkan 등...

표준 라이브러리

- 표준 라이브러리: C언어 표준에 명시된 기본 라이브러리
- C언어 표준에는 명세만 나와 있고 구현은 개발자의 몫
  - glibc, musl libc, mingw 등이 있음

표준 라이브러리

- printf, scanf도 표준 라이브러리의 일부
- 수학 연산, 입출력, 메모리 관리 등

math.h

- int abs(int n) – 절대값
- double pow(double base, double exponent) – b의 e제곱
- double sqrt(double arg) – 제곱근
- double log(double arg) – 로그
- double sin(double arg), cos, tan, asin, acos, atan, sinh, cosh, tanh, asinh, acosh, atanh

stdio.h

- FILE fopen(cost char *filename, const char *mode) – 파일 열기
- int fclose(FILE *stream) – 파일 닫기
- size_t fread(void *buffer, size_t size, size_t count, FILE *stream) – 파일 읽기
- size_t fwrite(cosnt void *buffer, size_t size, size_t count, FILE *stream) – 파일 쓰기

stdlib.h

- void exit(int exit_code) – 프로그램 종료
- void system(cosnt char *command) – 명령 실행

C Standard Library header files

The interface of C standard library is defined by the following collection of headers.

`<assert.h>`	Conditionally compiled macro that compares its argument to zero
`<complex.h>` (since C99)	Complex number arithmetic
`<ctype.h>`	Functions to determine the type contained in character data
`<errno.h>`	Macros reporting error conditions
`<fenv.h>` (since C99)	Floating-point environment
`<float.h>`	Limits of floating-point types
`<inttypes.h>` (since C99)	Format conversion of integer types
`<iso646.h>` (since C95)	Alternative operator spellings
`<limits.h>`	Ranges of integer types
`<locale.h>`	Localization utilities
`<math.h>`	Common mathematics functions
`<setjmp.h>`	Nonlocal jumps
`<signal.h>`	Signal handling
`<stdalign.h>` (since C11)	`alignas` and `alignof` convenience macros
`<stdarg.h>`	Variable arguments
`<stdatomic.h>` (since C11)	Atomic operations
`<stdbit.h>` (since C23)	Macros to work with the byte and bit representations of types
`<stdbool.h>` (since C99)	Macros for boolean type
`<stdckdint.h>` (since C23)	macros for performing checked integer arithmetic
`<stddef.h>`	Common macro definitions
`<stdint.h>` (since C99)	Fixed-width integer types
`<stdio.h>`	Input/output
`<stdlib.h>`	General utilities: memory management, program utilities, string conversions, random numbers, algorithms
`<stdnoreturn.h>` (since C11)	`noreturn` convenience macro
`<string.h>`	String handling
`<tgmath.h>` (since C99)	Type-generic math (macros wrapping math.h and complex.h)
`<threads.h>` (since C11)	Thread library
`<time.h>`	Time/date utilities
`<uchar.h>` (since C11)	UTF-16 and UTF-32 character utilities
`<wchar.h>` (since C95)	Extended multibyte and wide character utilities
`<wctype.h>` (since C95)	Functions to determine the type contained in wide character data

*https://en.cppreference.com/w/c/header

과제

새싹 – 조건, 반복, 함수

문제 번호	제목	문제 번호	제목
1330	두 수 비교하기	10950	A+B - 3
9498	시험 성적	10952	A+B - 5
14681	사분면 고르기	2739	구구단
2753	윤년	2438	별 찍기 - 1
2420	사파리월드	10951	A+B - 4
2741	N 찍기	15964	이상한 기호
10872	팩토리얼	2475	검증수

‘https://noj.am/(문제 번호)’ 로 문제 페이지를 바로 열 수 있음

감사합니다