C 언어 치트시트

1. 기본 구조 및 컴파일

C 프로그램은 헤더 파일 포함, `main` 함수, 그리고 다양한 구문으로 구성됩니다.

```c
#include <stdio.h> // 표준 입출력 라이브러리

// 프로그램의 시작점
int main() {
    printf("Hello, World!\n");
    return 0;
}
```

- 컴파일 및 실행:

```
gcc -o hello hello.c
./hello
```

2. 변수와 자료형

- 기본 자료형:
  - `int`: 정수
  - `float`, `double`: 실수
  - `char`: 단일 문자
  - `void`: 타입이 없음을 명시
- 한정자: `short`, `long`, `signed`, `unsigned`
- 변수 선언: `type variable_name;` (예: `int age;`)
- 상수: `const double PI = 3.14;`
- 배열: `int numbers[10];`
- 문자열: `char greeting[] = "Hello";` (널 문자로 끝나는 문자 배열)

3. 연산자

- 산술: `+`, `-`, `*`, `/`, `%`, `++`, `--`
- 관계: `==`, `!=`, `>`, `<`, `>=`, `<=`
- 논리: `&&` (AND), `||` (OR), `!` (NOT)
- 비트: `&`, `|`, `^`, `~`, `<<`, `>>`
- 할당: `=`, `+=`, `-=`, `*=`, `/=`
- 기타: `sizeof`, `&` (주소), `*` (포인터), `?:` (삼항 연산자)

4. 제어 흐름

- `if-else`:

```c
if (condition) {
    // ...
} else if (condition) {
    // ...
} else {
    // ...
}
```

- `switch`:

```c
switch (variable) {
    case value1:
        // ...
        break;
    case value2:
        // ...
        break;
    default:
        // ...
}
```

- `for` 루프: `for (init; condition; increment) { ... }`
- `while` 루프: `while (condition) { ... }`
- `do-while` 루프: `do { ... } while (condition);`
- 루프 제어: `break`, `continue`

5. 함수

- 함수 선언 (Prototype): `return_type function_name(type param1, type param2);`
- 함수 정의:

```c
return_type function_name(type param1, type param2) {
    // 함수 본문
    return value;
}
```

6. 포인터

변수의 메모리 주소를 저장하는 변수.

- 선언: `type *pointer_name;` (예: `int *ptr;`)
- 주소 연산자 (`&`): 변수의 주소를 가져옴. `ptr = &my_var;`
- 간접 참조 연산자 (`*`): 포인터가 가리키는 주소의 값을 가져옴. `value = *ptr;`

7. 구조체 (Structs)

서로 다른 타입의 변수들을 하나의 단위로 묶음.

```c
struct Person {
    char name[50];
    int age;
};

// 사용
struct Person p1;
strcpy(p1.name, "Jules");
p1.age = 30;
```

- `typedef`: 새로운 타입 이름 정의. `typedef struct Person Person;`

8. 전처리기 (Preprocessor)

컴파일 전에 코드를 처리.

- `#include <filename>`: 시스템 헤더 파일 포함.
- `#include "filename"`: 사용자 정의 헤더 파일 포함.
- `#define NAME value`: 매크로 상수 정의.
- `#ifdef`, `#ifndef`, `#if`, `#else`, `#elif`, `#endif`: 조건부 컴파일.

9. 동적 메모리 할당

`#include <stdlib.h>` 필요.

- `malloc(size)`: 지정된 바이트 크기만큼 메모리 할당.
- `calloc(num, size)`: `num * size` 바이트만큼 할당하고 0으로 초기화.
- `realloc(ptr, size)`: 이미 할당된 메모리 블록의 크기 변경.
- `free(ptr)`: 할당된 메모리 해제.

10. 파일 입출력

`#include <stdio.h>` 필요.

- `FILE *fp;`: 파일 포인터.
- `fopen("filename", "mode")`: 파일 열기. 모드: `r`(읽기), `w`(쓰기), `a`(추가).
- `fclose(fp)`: 파일 닫기.
- `fprintf(fp, "...", ...)`: 파일에 형식화된 출력 쓰기.
- `fscanf(fp, "...", ...)`: 파일에서 형식화된 입력 읽기.
- `fgetc(fp)`, `fputc(c, fp)`: 문자 단위 입출력.
- `fgets(str, n, fp)`, `fputs(str, fp)`: 문자열 단위 입출력.