Basic_Concepts_3 - Pointer Operations

Pointer Operations

포인터 `T*`는 메모리에 있는 공간을 참조하는 값이다.

포인터 역참조`(*ptr)`는 포인터가 참조하고 있는 공간에 저장된 값을 얻는것을 의미한다.

subscript 연산자`(ptr[])`는 포인터의 전달받은 위치에 해당하는 원소에 접근하는것을 허가한다.

포인터의 타입은 32비트/64비트의 부호가 없는 정수형이다. 비트의 크기는 아키텍쳐에 따라 다르다.

- 오직 `+`, `-`, `++`, `--`, 비교 연산자(`==`, `!=`, `<`, `<=`, `>`, `>=`), subscript `[]`, 역참조 `*` 연산자만 지원한다.
- 포인터는 명시적 형변환을 통해 정수형 타입으로 변환할 수 있다.

```
void* x;
size_t y = (size_t)x; // ok
//size_t y = x; // 컴파일 에러
```

어떤 포인터든 `void*` 타입으로 암시적 형변환이 가능하다.

void 타입이 아닌 포인터 타입으로 형변환할땐 반드시 명시적 형변환을 해야한다.

`static_cast`로는 포인터타입을 안전하게 형변환 할 수 없다. `void*` 예외가 발생하기 때문이다.

→ 자세한 이유는 static_cast를 다루는 강의에서 확인할 수 있다.

```
int* ptr1 = ...;
void* ptr2 = ptr1; // int* -> void* 암시적 형변환이 일어난다.

void* ptr3 = ...;
int* ptr4 = (int*)ptr3; // void* -> int* 명시적 형변환이 일어난다.
// static_cast를 통해 형변환이 가능하다.

int* ptr5 = ...;
```

```
char* ptr6 = (char*)ptr5; // int* -> char* 명시적 형변환이 일어난다
// static_cast를 통해 형변환할 수 없다. 위험하다.
```

일반적인 실수

```
int* ptr1, ptr2; // 하나는 포인터 타입이지만, 하나는 정수형 타입이다.
itn* ptr3, *ptr4; // 둘다 포인터 타입이다.
```