6주차

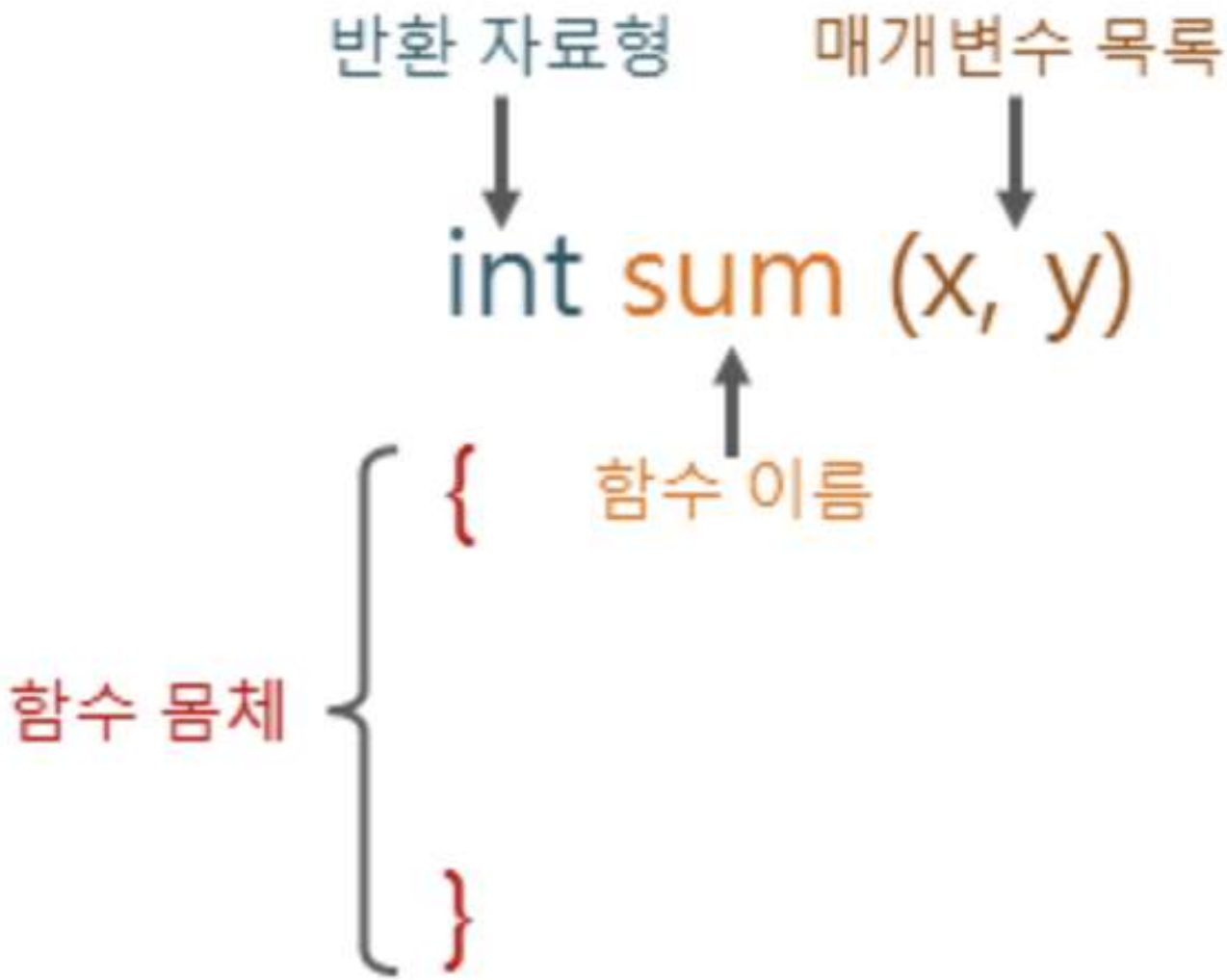
함수란?

- y = f(x)

-> x input에 대해 y output이 나옴

- 매개변수(parameter)를 받아 값을 반환(return)하는 것

- Function



예시

```c
#include "stdio.h"
#include "string.h"

void print_ee(void) {

    printf("ee \n");

    return;
}

int my_add(int a, int b) {

    return a + b;
}

int main(void) {

    print_ee();

    printf("%d \n", my_add(3, 4));

    return 0;
}

```

포인터와 주소

- 가리키는 것
- 주소 연산자 &, 형식 지정자는 %p
- P는 a를 참조
- *P는 a를 역참조
- Adress, reference, dereference

```c
#include "stdio.h"
#include "string.h"

int main(void) {

    int a = 4;
    int* location_of_a = &a;

    printf("%d \n", *location_of_a);
    printf("%p \n", location_of_a);

    return 0;
}

```

구조체란?

- 여러 자료의 집합
- 하나의 자료형처럼 사용 가능
- Structure

```c
#include "stdio.h"
#include "string.h"

struct coordinate {
    int y;
    int x;
};

int main(void) {

    coordinate coo[4] = {{3, 4}, {1, 6}, {-1, 4}};

    for (int i = 0; i < 4; ++i) {
        printf("%d %d \n", coo[i].y, coo[i].x);
    }

    return 0;
}

```

시간복잡도란?

- Big-O notation에 의한 시간 계산 법 ex. O(n), O(n^2)…
- Worst case를 상정하여
- 해당 코드를 test하는 척도 중 하나

- Ex. 2차원 배열의 for문 일반 순회

- Time complexity

```
21
22  int sero, garo;
23  vector<vector<int>> grid;
24
25  int main() {
26      ios::sync_with_stdio(false);
27      cin.tie(nullptr);
28
29      sero = 432;
30      garo = 0x43;
31
32      grid.resize(sero, vector<int>(garo));
33
34      for (int i = 0; i < sero; ++i) {
35          for (int j = 0; j < garo; ++j) {
36              // somehow exploration about grid
37          }
38      }
39
40      return 0;
41  }
42
```

함수 작동 원리

- 코드 영역(Text) : 함수 기계어 명령
- 데이터 영역(Data) : 전역/정적 변수
- 힙(Heap) : malloc/free (동적 할당)
- 스택(Stack) : 함수 호출 시 생성되는 프레임

함수 작동 원리

1. 프로그램 진행 중 해당 함수로 제어권 양도
2. 기존의 진행점으로 돌아올 수 있도록 주소 기억
3. 호출된 함수의 인자 평가(파라미터 6 이상이면 레지스터가 아닌 스택에서 저장)
4. 호출된 함수를 위한 스택프레임 공간 확보
5. 본문 실행
6. 사용한 스택 정리
7. 리턴값 획득

함수 호출 비용

- 함수를 이용하면 리펙토링과 유지보수성에 있어 뛰어남
- But, 각각의 n회 실행에 대해 상수배로 러닝타임이 늘어날 수 있음
- 시간초과가 날 때 정량적인 비교를 통해 함수를 다른 방식으로 구현하면 맞을 수도 있음

CPU 연산시간 concept

- 대충 1억번 == 1초
- 컴퓨터마다 당연히 다 다르지만 백준 온라인 컴파일러와 표준은 1억번의 CPU 연산을 1초로 규정
- 로컬 환경에서 되는데 백준에서 안 된다~ 하면 안 됨
- 파이썬은 당연히 훨씬 더 느림

함수의 매개변수 전달 방법

1. 복사본을 전달

2. 주소를 전달하여 역참조

3. 동일한 방법으로 배열에 적용 가능

매개변수 전달 – 복사본

```c
#include "stdio.h"
#include "string.h"

void exx(int a) {
    a++;
    return;
}

int main(void) {

    int a = 4;
    int b = 4;

    exx(a);

    if (a == b) {
        printf("a is same as b \n");
    } else {
        printf("a is different from b \n");
    }

    return 0;
}
```

매개변수 전달 – 주소

```c
#include "stdio.h"
#include "string.h"

void exx(int* a) {
    (*a)++;
    return;
}

int main(void) {

    int a = 4;
    int b = 4;

    exx(&a);

    if (a == b) {
        printf("a is same as b \n");
    } else {
        printf("a is different from b \n");
    }

    return 0;
}

```

주의점
– 스택프레임 소멸

```c
#include <stdio.h>

int *gptr;

void exx(int a) {
    gptr = &a;

    return;
}

int main(void) {

    int a = 4;
    int b = 6;

    exx(a);

    printf("%d %d %d \n", a, b, *gptr);

    return 0;
}
```