

게임자료구조와알고리즘 -CHAPTER4-

SOULSEEK



목차

- 1. 함수 고급 (디폴트 매개변수)
- **2.** 함수 고급 **(**오버로딩**)**
- **3.** 함수 고급 (함수 포인터)
- **4.** 함수 고급 (재귀호출)





• 함수 호출 시 필요한 매개변수에 인자 값을 전달하지 않을 시 자동으로 기본 값을 적용하게 하는 변수

```
void SetFramesPerSec(int fps = 60)
{
    cout << "FPS = " << fps << endl;
}

void main()
{
    SetFramesPerSec();
    //SetFramesPerSec(10);
}</pre>
```

```
void func(int n = 10)
   int Sum = 0;
   for (int i = 1; i <= n; i++)
   Sum += i;
   cout << "1 ~ " << n << "까지의 총 합 : " << Sum << endl;
void main()
   func(100);
   func();
```

```
1. 함수 고급 (디폴트 매개변수)
void Addfunc(int& x, int y = 2)
   int count = 0;
   while (count < y)
       x *= x;
       count++;
void main()
   int a;
   cout << "입력값:";
   cin >> a;
   Addfunc(a);
   //Addfunc(a, 10);
   cout << "출력값 : " << a << endl;
```

```
Info userInfo = { 0 };
struct Item
                            void UserInfo(char nick_[12], int level_, int hp_, int mp_, Item item_ = { 0 })
    int dmg;
                                 userInfo.level = level ;
    int str;
                                 userInfo.hp = hp_;
    int min;
                                 userInfo.mp = mp_;
                                 userInfo.item = item_;
};
                                 cout << "======User Info=======" << endl;
struct Info
                                 cout << "level : " << userInfo.level << endl;</pre>
                                 cout << "hp : " << userInfo.hp << endl;</pre>
                                 cout << "mp : " << userInfo.mp << endl;</pre>
    int level;
    int hp;
    int mp;
                            void main()
     Item item;
                                 UserInfo(11, 1000, 1000);
};
                                 //UserInfo(11, 1000, 1000, item);
```

학습과제

- 1부터 입력 받은 숫자까지의 합을 구해보자.(기본값을 설정하고 문자열이 입력되면 기본값으로 숫자일 경우는 입력된 수 까지로)
- 두개의 숫자를 입력 받아 두 수 사이에 있는 구구단을 출력하자(2,5가 입력되면 2단에서 5단까지 출력,문자열이 입력된 경우 디폴트로 2~9단까지 출력하게 하여라)



함수 고급 (오버로딩)

2. 함수 고급 (오버로딩)

- 여러 함수들이 동일한 이름을 사용할 수 있는 기능을 말한다.
- 함수의 인자(시그니처)를 이용해 가장 어울리는 인자타입을 가진 선정해 준다.
- 반환 값만 틀린 경우는 오버로드 할 수 없다.
- 시그니처가 다르더라도 레퍼런스 같은 경우 오버로드 할 수 없다.
- 애매한 규칙은 나름대로의 규칙을 가지고 알아서 선정해 버리기 때문에 확실하게 사용하자.

```
void func()
   cout << "func1() 함수를 호출하였습니다." << endl;
void func(int a)
   cout << "func1(int a) 함수를 호출하였습니다." << endl;
void main()
   func();
   func(10);
```

2. 함수 고급 (오버로딩)

```
struct Info
     int level;
     int hp;
     int mp;
};
Info userInfo = { 0 };
void UserInfo(int level_)
     userInfo.level = level;
     cout << "======User Info=======" << endl:
     cout << "level : " << userInfo.level << endl;</pre>
     cout << "hp : " << userInfo.hp << endl;</pre>
     cout << "mp : " << userInfo.mp << endl;</pre>
```

```
void UserInfo(int hp_, int mp_)
{
    userInfo.hp = hp_;
    userInfo.mp = mp_;

    cout << "======User Info=======" << endl;
    cout << "level : " << userInfo.level << endl;
    cout << "hp : " << userInfo.hp << endl;
    cout << "mp : " << userInfo.mp << endl;
}

void main()
{
    UserInfo(11);
    UserInfo(1000, 1000);
}</pre>
```

2. 함수 고급 (오버로딩)

학습과제

- 두 정수를 입력 받아서 1부터 뒤에 입력한 수까지 앞의 수의 배수들의 합을 구하는 함수를 만들고 문자와 정수를 받게 되면 문자열을 정수만큼 출력하는 함수를 만들어 보자.
- 5개의 문자를 입력 받으면 내림차순으로 5개의 숫자를 입력 받으면 오름차순으로 정렬하는 함수를 만들어 보자.



함수 고급 (함수 포인터)

3. 함수 고급 (함수 포인터)

• 함수의 주소를 받는 변수를 만들어 함수를 변수처럼 활용할 수 있다.

```
void func1()
   cout << "함수 포인터1 호출" << endl;
void func2()
   cout << "함수 포인터2 호출" << endl;
void main()
   void(*p) ();
   p = &func1;
   p();
   p = &func2;
   p();
   return;
```

```
3. 함수 고급 (함수 포인터)
void func1(int x, int y)
   cout << x << " + " << y << " = " << x + y << endl;
void func2(void(*p)(int x, int y))
   (*p)(10, 15);
void main()
   void(*p) (int x, int y);
   p = &func1;
   func2(p);
   return;
```

```
3. 함수 고급 (함수 포인터)
typedef void(*FUNC)(int, int);
void func1(int x, int y)
   cout << x << " + " << y << " = " << x + y << endl;
void func2(FUNC p)
   p(10, 15);
void main()
   FUNC p;
   p = &func1;
   func2(p);
   return;
```



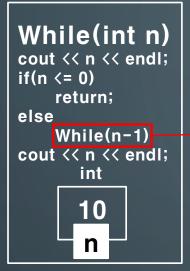
함수 고급 (재귀 함수)

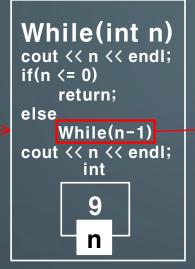
3. 함수 고급 (재귀 함수)

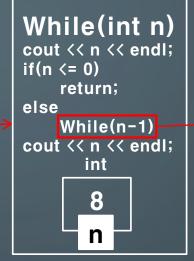
- 함수 내부에서 자신을 호출함으로써 함수 호출을 반복하는 작업
- 재귀가 풀리기 위한 조건을 걸지 않을 시 무한루프에 빠질 수 있으므로 조건을 신경 써서 잘 설계 하여야 한다.

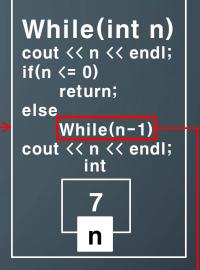
```
void While(int n)
   cout << "n = " << n << endl;
   if (n \le 0)
       return;
   else
       While(n - 1);
   cout << "n = " << n << endl;
void main()
   While(10);
   return;
```

3. 함수 고급 (재귀 함수)



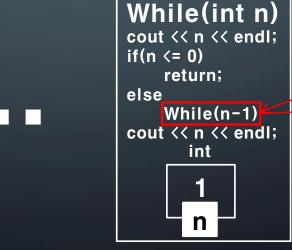


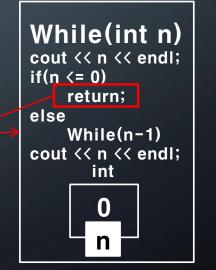




```
While(int n)
cout << n << endl;
if(n \le 0)
    return;
else
    While(n-1)
cout << n << endl;
       int
```







3. 함수 고급 (재귀 함수)

학습과제

- 재귀 함수를 이용하여 정수를 입력 받고 2진수로 변환하여 출력하자.
- 재귀 함수를 이용하여 1부터 원하는 수까지의 합을 구해보자.
- 재귀 함수를 이용하여 입력 받은 수의 팩토리얼을 구해보자.

SetConsoleCursorPostion API

- Console용 winAPI함수이다.
- 커서의 위치는 원하는 곳으로 이동시켜주는 함수이다.
- 커서의 위치에 있는 문자만 다시 그리기 위해 사용된다.

좌표를 입력 받아서 커서의 위치를 옮겨주는 알고리즘 구현

```
void gotoxy(int x,int y)
{
    COORD Pos; // 좌표 값을 설정할 수 있는 구조체
    Pos.X=x;
    Pos.Y=y;

SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE),Pos);
}
```

```
게임제작에 필요한 기능
#include <windows.h>
void gotoxy(int x,int y);
void main()
  int cnt=-1;
  char *str="2016/11/13!";
  cout << "HelloWorld!";</pre>
  while(cnt++!=12)
    gotoxy(cnt, 0);
     cout << str[cnt];</pre>
     Sleep(500);
void gotoxy(int x,int y)
  COORD Pos;
  Pos.X=x;
  Pos.Y=y;
  SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE),Pos);
```

```
#include<iostream>
#include<time.h>
#include<Windows.h>
using namespace std;
void gotoxy(int x, int y)
    COORD Pos = { x, y };
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), Pos);
void main()
    int OldClock, CurClock, i = 0;
    OldClock = clock();
    while (1)
         CurClock = clock();
         if (CurClock - OldClock >= 1000)
             į++;
              gotoxy(10, 10);
              cout << "Count :" << left << i;
              OldClock = CurClock;
```

```
void gotoxy(int x, int y)
     COORD Pos = { x, y };
     SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), Pos);
void main()
     int CountClock, SayClock, CurClock, i = 0, j = 0;
     CountClock = clock();
     SayClock = clock();
     while (1)
           CurClock = clock();
          if (CurClock - CountClock >= 1000)
                i++;
                gotoxy(10, 10);
                cout << "Count :" << i;
                CountClock = CurClock;
          if (CurClock - SayClock >= 5000)
                j++;
                gotoxy(10, 11 + j);
                cout << j << "번째 인사 : Hello~";
                SayClock = CurClock;
```

학습과제

- 타이머를 만들어 보자.
- 문자배열을 하나 정하고 해당 문자의 색깔을 하나씩 순차적으로 계속 변화시켜서 무지개 효과를 만들어 보자.

게임제작 과제

• 콘솔용 지뢰 찾기를 만들어 보자.