4강. 함수와 라이브러리 함수

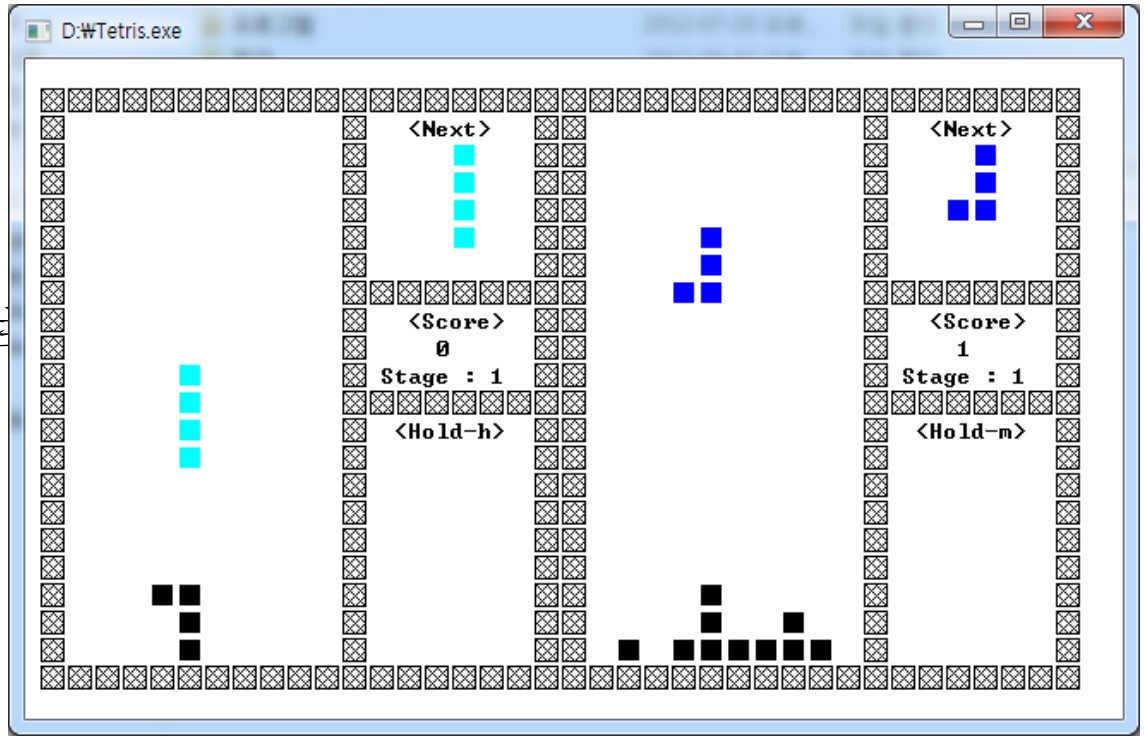
C++ 프로그래밍

jhhwang@kumoh.ac.kr

컴퓨터공학과 황준하

목차

- 함수 오버로딩
- 디폴트 매개변수
- 라이브러리 함수
  - clock 함수
  - 난수 발생
  - 비버퍼형 문자 입력
  - 커서 이동
- 프로그래밍 문제



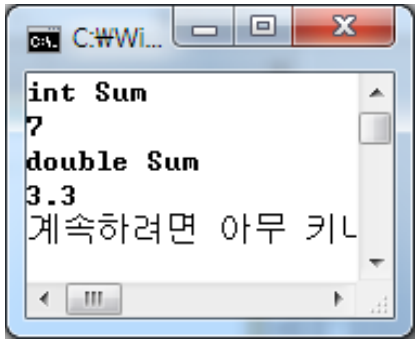
함수 오버로딩

- 함수 오버로딩
  - 동일한 이름의 함수를 여러 개 만들 수 있음
  - 함수 프로파일이 달라야 함
    - 함수 프로파일 = 함수명 + 매개변수의 개수 + 매개변수의 타입

```
int Sum(int x, int y)
{
	cout << "int Sum" << endl;
	int z = x + y;
	return z;
}
```

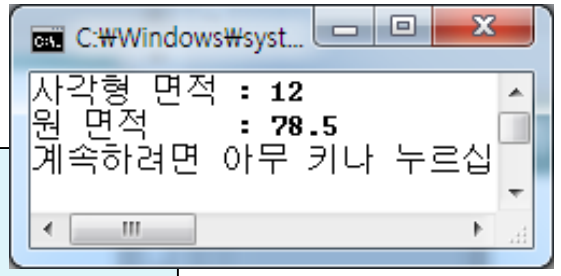
```
void main(void)
{
	cout << Sum(3, 4) << endl;
	cout << Sum(1.1, 2.2) << endl;
}
```

```
double Sum(double x, double y)
{
	cout << "double Sum" << endl;
	double z = x + y;
	return z;
}
```



함수 오버로딩 : 연습 문제

- 사각형의 면적, 원의 면적을 구하는 함수 GetArea를 각각 만들고 main 함수와 같이 테스트해 보라.
  - 사각형의 입력값 : 가로, 세로
  - 원의 입력각 : 반지름
  - 모두 double 값으로 처리하라.




```
void main(void)
{
	cout << "사각형 면적 : " << GetArea(3, 4) << endl;
	cout << "원 면적 : " << GetArea(5) << endl;
}
```

함수 오버로딩 : 연습 문제

- 프로그램 확인

```
double GetArea(double Garo, double Sero)
{
    return (Garo * Sero);
}

double GetArea(double Radius)
{
    return (3.14 * Radius * Radius);
}
```

디폴트 매개변수

- 디폴트 매개변수
  - 대응되는 매개변수의 값이 입력되지 않을 경우 디폴트 값 배정

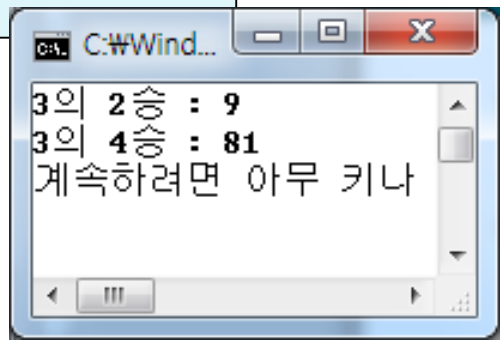
```
int Sum(int Num1, int Num2, int Num3 = 1, int Num4 = 2, int Num5 = 3);

void main(void)
{
	cout << Sum(1, 2) << endl;
	cout << Sum(1, 2, 3) << endl;
	cout << Sum(1, 2, 3, 4) << endl;
	cout << Sum(1, 2, 3, 4, 5) << endl;
}

int Sum(int Num1, int Num2, int Num3, int Num4, int Num5)
{
	return (Num1 + Num2 + Num3 + Num4 + Num5);
}
```

값이 넘어오지 않을 경우 디폴트값

디폴트 매개변수 : 연습 문제

- 정수값 x의 y승을 구하는 power 함수를 만들어 보라. 단, y 값이 전달되지 않을 경우 x의 2승을 반환하라.
  - 디폴트 매개변수를 사용하여 하나의 함수만 작성하라.

```
void main(void)
{
	cout << "3의 2승 : " << power(3) << endl;
	cout << "3의 4승 : " << power(3, 4) << endl;
}
```



디폴트 매개변수 : 연습 문제

- 프로그램 확인

```
int power(int x, int y = 2)
{
	int result = 1;

	for (int i = 0; i < y; i++)
		result *= x;

	return result;
}
```

라이브러리 함수

- 라이브러리
  - 미리 만들어 놓은 함수, 클래스의 집합
- C++ 프로그램 제작
  - 본인이 직접 함수와 클래스 작성 : 50%
  - 미리 준비되어 있는 라이브러리 활용 : 50%
- C++ 표준 라이브러리
  - C++ 표준에서 기술되어 있는 라이브러리의 집합
    - 함수
    - 클래스
    - 템플릿 등

라이브러리 함수

- 여기서 소개할 라이브러리 함수

함수명	헤더 파일	용도	표준/비표준
clock	<ctime>	clock 기준 경과 시간	표준
rand	<cstdlib>	임의의 난수 발생	표준
getch	<conio.h>	비버퍼형 문자 입력	비표준
SetConsoleCursorPosition	<windows.h>	실행 도스창의 커서 위치 변경	비표준

clock 함수

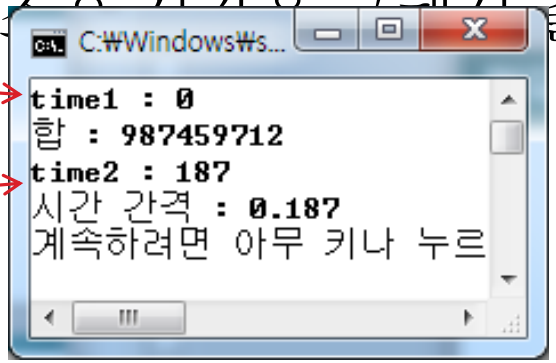
- clock 함수
  - 프로그램 실행 시 내부적인 clock 값이 0으로 초기화됨
  - 이 값은 초당 일정 횟수만큼(CLOCKS_PER_SEC, 1000) 증가

→ 특정 시각 사이의 경과 시간을 알아오는 활용

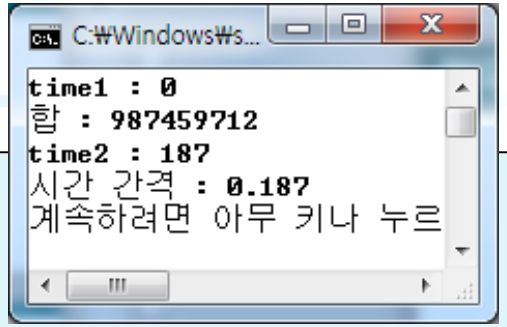
- 연습 문제
  - 1부터 1억까지의 합을 구하되 계산 소요 시간을 구해서 출력해 보세요.

합산 시작 시 clock 값

합산 종료 시 clock 값



clock 함수




```
#include <iostream>
#include <ctime>
using namespace std;

void main(void)
{
	clock_t time1 = clock();
	cout << "time1 : " << time1 << endl;

	int Sum = 0;
	for (int i = 1; i <= 100000000; i++) {
		Sum += i;
	}

	cout << "합 : " << Sum << endl;

	clock_t time2 = clock();
	cout << "time2 : " << time2 << endl;
	cout << "시간 간격 : " << (double) (time2 - time1) / CLOCKS_PER_SEC << endl;
}
```

현재 clock 수는 clock_t 타입 변수에 저장

clock_t는 int와 동일

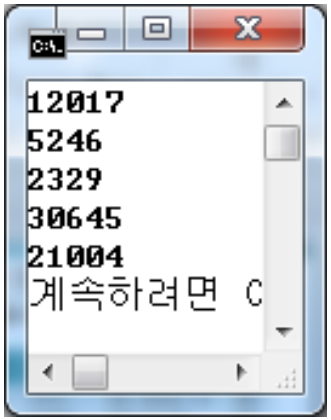
초 단위 시간 간격 구하기

난수 발생 : rand 함수

- rand 함수 : 임의의 난수 발생
  - 난수의 범위 : 0 ~ RAND_MAX (0x7FFF)
  - srand 함수를 통해 난수 발생기를 초기화해야 함
    - srand(정수값) : 정수값이 동일하면 매번 생성되는 난수 순서 동일

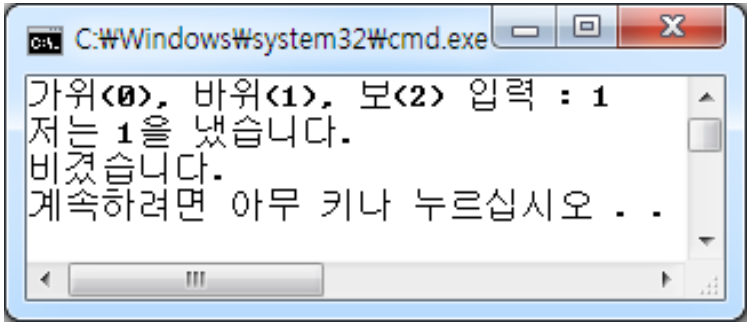
```
#include <iostream>
#include <ctime>          // time 함수
#include <cstdlib>        // srand, rand 함수
using namespace std;

void main(void)
{
      srand(time(NULL));

      for (int i = 0; i < 5; i++)
            cout << rand() << endl;
}
```

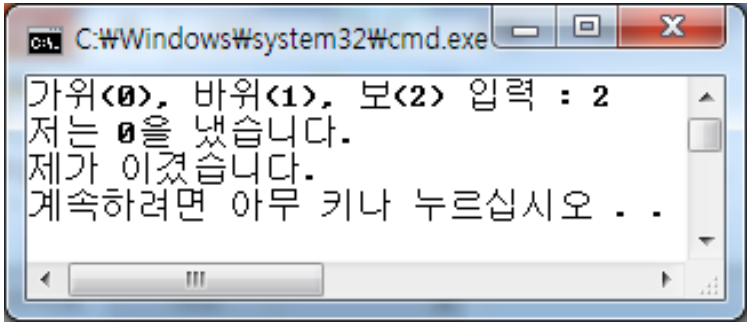
그램 실행 시 마다 다름



난수 발생 : 연습 문제

- 가위, 바위, 보 게임을 만든다.
  - 사용자는 가위, 바위, 보 중 하나를 선택하도록 하고
  - 컴퓨터는 rand 함수를 사용하여 무작위로 내도록 한다.
  - 매번 결과가 다르도록 srand 함수를 사용하라.





난수 발생 : (

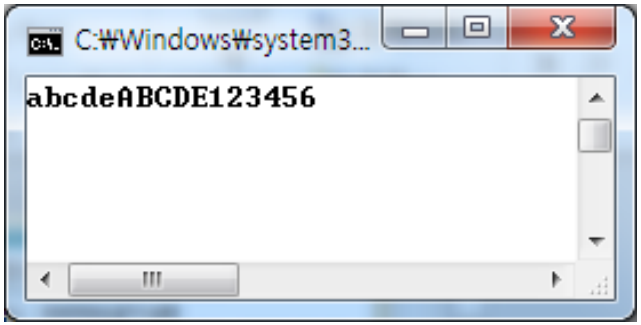
- 프로그램 확인

```
void main(void)
{
	srand(time(NULL));
	int Com, Human;

	cout << "가위(0), 바위(1), 보(2) 입력 : ";
	cin >> Human;

	Com = rand() % 3;
	cout << "저는 " << Com << "을 냈습니다." << endl;

	switch (Com) {
	case 0 :
		if (Human == 0)
			cout << "비겼습니다." << endl;
		else if (Human == 1)
			cout << "당신이 이겼습니다." << endl;
		else
			cout << "제가 이겼습니다." << endl;
		break;
	case 1 :
		// 생략
	case 2 :
		// 생략
	}
}
```

비버퍼형 문자 입력 : getch 함수

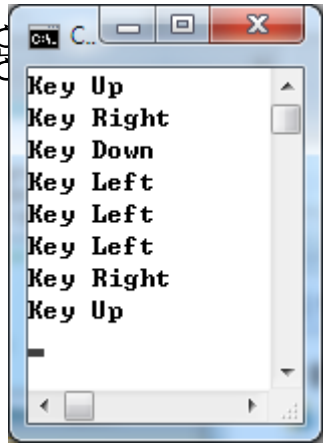
- getch 함수 : 비버퍼형 문자 입력
  - cin ›› ch : 입력 문자가 화면에 나타남. Enter 입력 후 실제 실행
  - [illegible]타나지 않음. 바로 실행

```
#include <iostream>
#include <conio.h>     // getch 함수
using namespace std;

void main(void)
{
      int ch;

      while ((ch = getch()) != 'q') {
            cout << (char) ch;
      }
}
```



getch 함수 활용

- 방향키를 비롯하여 현재 입력한 키가 무엇인지 알아내는 함수 GetKey를 만들기
  - 게임 프로그램 제작 시 유용




```
void main(void)
{
	int ch;

	while ((ch = GetKey()) != KEY_ESC) {
		switch (ch) {
		case KEY_RIGHT :
			cout << "Key Right" << endl;
			break;
		case KEY_LEFT :
			cout << "Key Left" << endl;
			break;
		case KEY_UP :
			cout << "Key Up" << endl;
			break;
		case KEY_DOWN :
			cout << "Key Down" << endl;
			break;
		}
	}
}
```

getch 함수 활용

방향키 등 특수키에 해당하는 키값을 문자열 상수로 정의 (편의성)

UP Key의 경우 키값이 72지만 72 값을 갖는 'H' 문자와의 혼동을 피하기 위해 256을 더함

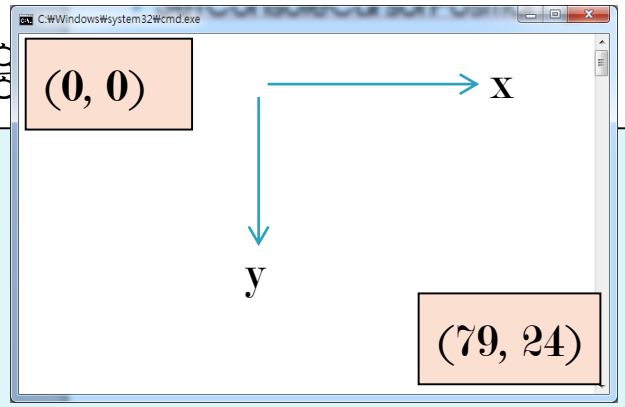
방향키의 경우 0 또는 224라는 값이 먼저 입력된 후 해당 방향키 값이 입력됨 (2자 입력)

```
#include <iostream>
#include <conio.h>   // getch 함수
using namespace std;

#define KEY_ESC 27
#define KEY_UP (256 + 72)
#define KEY_DOWN (256 + 80)
#define KEY_LEFT (256 + 75)
#define KEY_RIGHT (256 + 77)

int GetKey(void)
{
      int ch = getch();

      if (ch == 0 || ch == 224)
            ch = 256 + getch();

      return ch;
}
```

커서 이동 : SetConsoleCursorPosition 함수

- SetConsoleCursorPosition 함수
  - 실행 도스창의 커서 위치 이동
  - 게임 등 동적인 프로그램 제작 시 유용

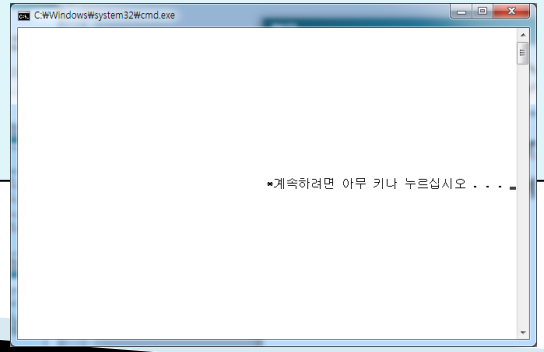
디폴트 크기 : (80, 25)




```
#include <iostream>
#include <windows.h>
using namespace std;

void main(void)
{
    COORD Pos = { 40, 12 };
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), Pos);

    cout << "*";
}
```

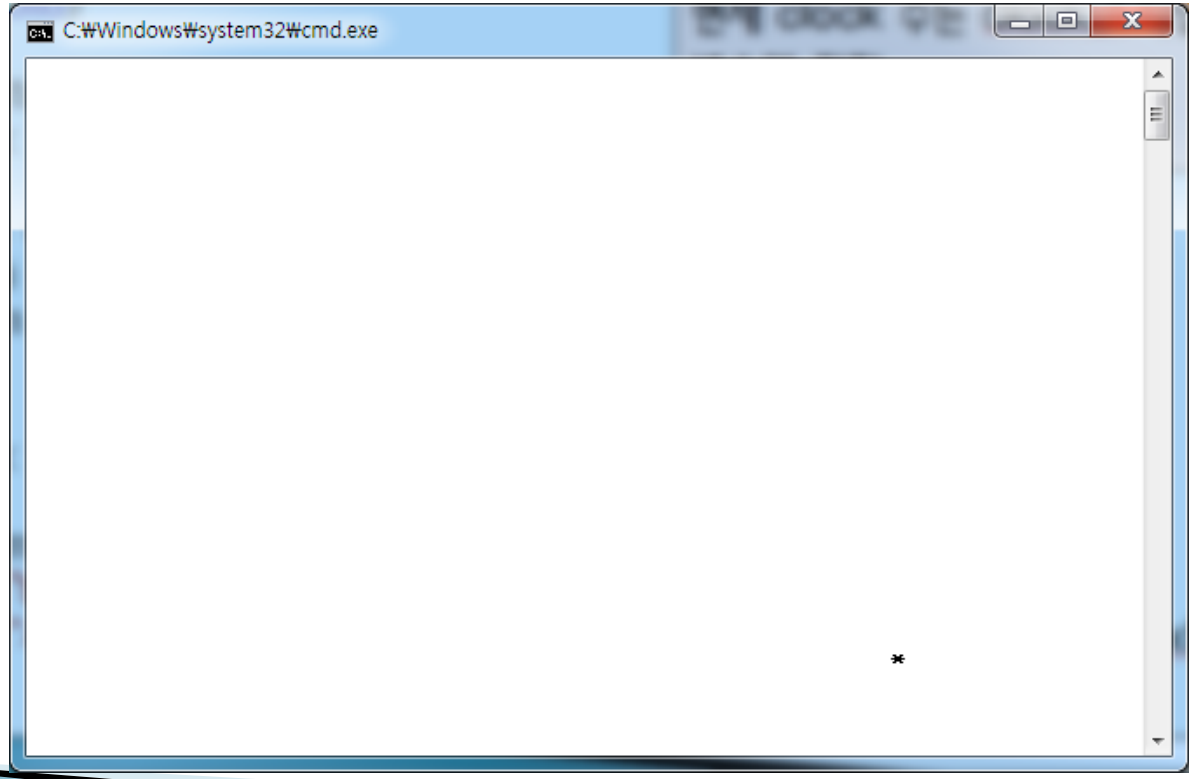
좌표를 나타내는 구조체

모니터 핸들



프로그래밍 문제

- 0.5초 간격으로 화면 내의((0, 0)~(79, 24)) 임의의 위치에 '*'가 나타나도록 만들어 보라.
  - 이전에 있던 *는 삭제되어야 한다.



프로그래밍 문제

- 프로그램 확인

```
#include <iostream>
#include <cstdlib>
#include <ctime>
#include <windows.h>
using namespace std;

void GotoXY(int x, int y) //커서 위치 이동
{
	COORD Pos = { x, y };
	SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), Pos);
}
```

프로

- 프로

```
void main(void)
{
	srand(time(NULL));
	clock_t time1 = clock();
	clock_t time2;
	int x = -1, y = -1;

	while (true) {
		time2 = clock();

		if (((double) (time2 - time1) / CLOCKS_PER_SEC) > 0.5) {
			if (x >= 0) {			// 현재 위치 삭제
				GotoXY(x, y);
				cout << " ";
			}

			x = rand() % 80;
			y = rand() % 25;
			GotoXY(x, y);

			cout << "*";
			time1 = clock();		// 기준 시간 초기화
		}
	}
}
```