

제3장 함수와 문자열

1. 함수의 기본적인 개념을 이해한다.
2. 인수와 매개 변수의 개념을 이해한다.
3. 함수의 인수 전달 방법 2가지를 이해한다
4. 중복 함수를 이해한다.
5. 디폴트 매개 변수를 이해한다.
6. 문자열의 구성을 이해한다.
7. string 클래스의 사용법을 익힌다.

1

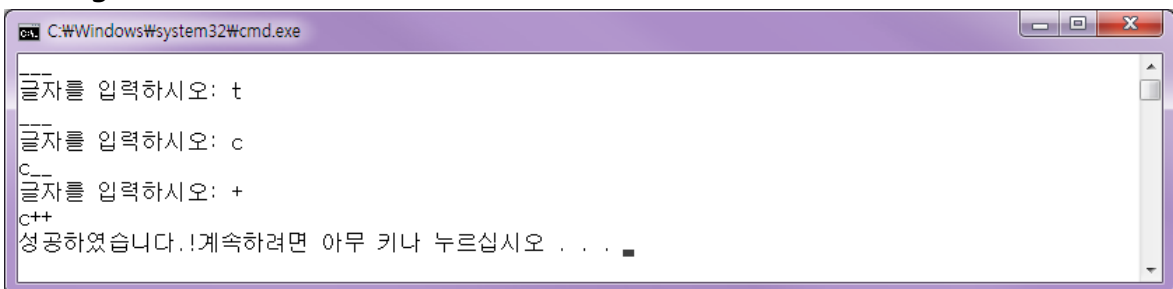
이번 장에서 만들어볼 프로그램

2개, 3개, 4개의 정수의 합을 계산하는 함수 **sum()**을 작성하여 사용



```
C:\Windows\system32\cmd.exe
sum(10, 15)=25
sum(10, 15, 25)=50
sum(10, 15, 25, 30)=80
계속하려면 아무 키나 누르십시오 . . .
```

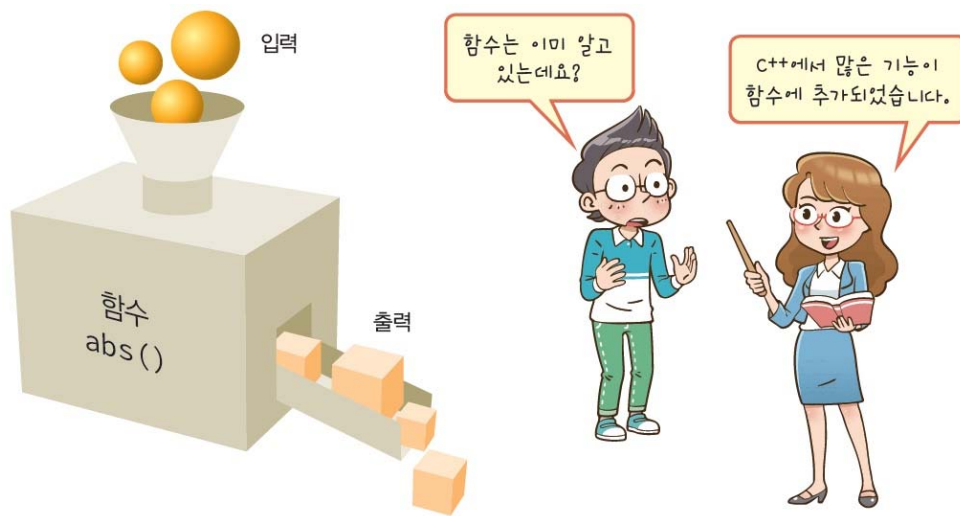
string 클래스를 이용한 행맨 게임



```
C:\Windows\system32\cmd.exe
---
글자를 입력하십시오: t
---
글자를 입력하십시오: c
C_--
글자를 입력하십시오: +
c++
성공하였습니다. !계속하려면 아무 키나 누르십시오 . . . ■
```

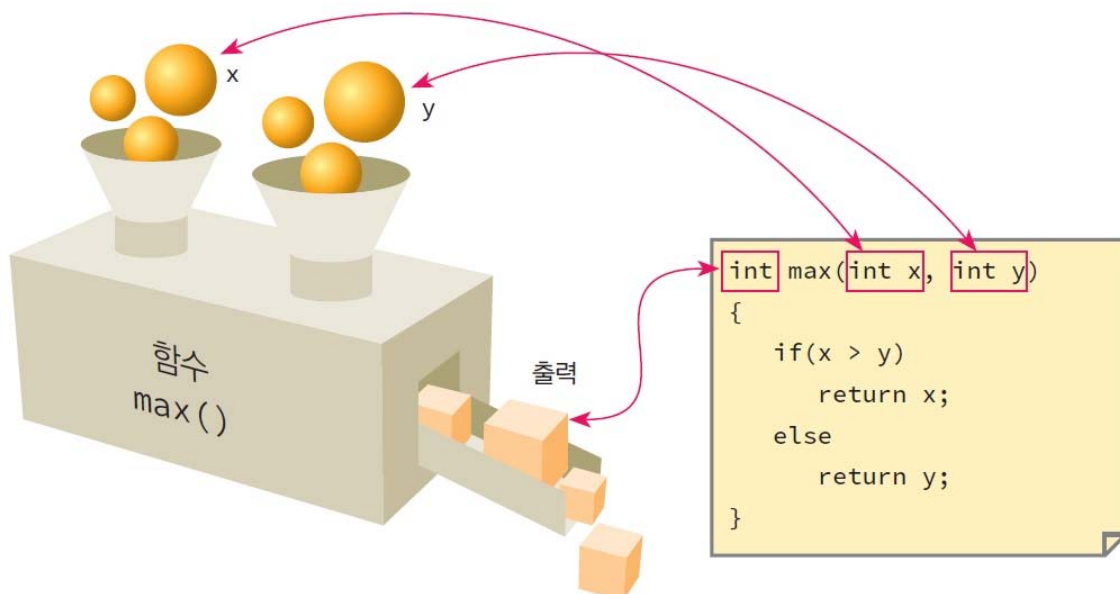
2

3.2 함수란?



3

함수 선언



4

함수 호출 (function call)

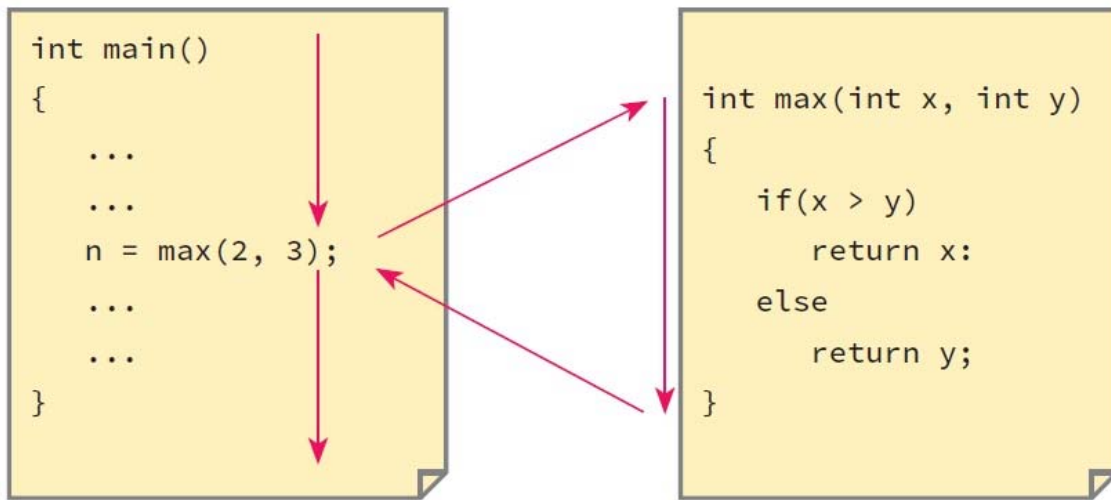


그림 3.1 함수 호출

5

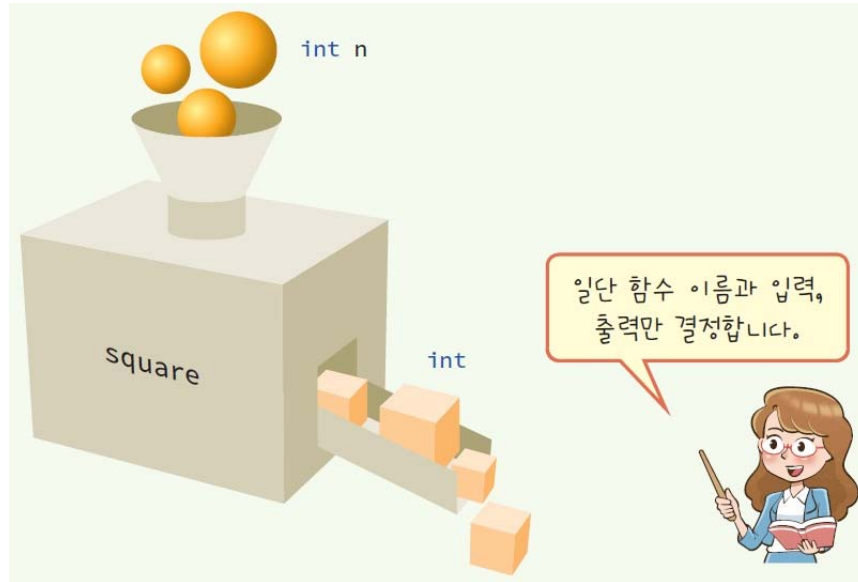
예제

```
#include <iostream>
using namespace std;
// 함수 정의
int max(int x, int y)
{
    if (x>y)
        return x;
    else
        return y;
}
int main()
{
    int n;
    n = max(2, 3);    // 함수 호출
    cout << "연산 결과 = " << n << endl;
    return 0;
}
```

6

Lab: 함수 만들기

- 정수의 제곱값을 구하는 함수를 만들어보자.



7

예제

```
#include <iostream>
using namespace std;
int square(int n)
{
    return(n*n);
}
int main()
{
    int n;
    cout << "제공할 정수를 입력하시오: ";
    cin >> n;

    cout << square(n) << endl;
    return 0;
}
```

8

실습

- 3개의 정수 중에서 최대값을 찾는 함수 `maximum(x,y,z)`를 정의하고 테스트하라.

-
- 0부터 9까지의 난수를 100번 생성하여 가장 많이 생성된 수를 출력하는 프로그램을 작성하라. 난수는 `rand()` 함수를 사용하여 생성하라.

3.3 함수 원형 정의

- 함수 원형(**function prototype**)은 함수의 이름, 매개변수, 반환형을 함수가 정의되기 전에 미리 한번 써주는 것이다.

```
#include <stdio.h>
int square(int n);
int main()
{
    int result;
    result = square(5);
    printf("%d \n", result);
}
int square(int n)
{
    return(n * n);
}
```

11

실습

- 정수를 1과 자신만으로 나눌 수 있다면 소수라고 한다. 예를 들어 2, 3, 5, 7은 소수이다. 주어진 숫자가 소수인지 여부를 결정하는 함수 **prime()**을 작성하라. 이 함수를 이용하여 2와 100사이의 모든 소수를 판별하고 출력하는 프로그램을 작성하라.

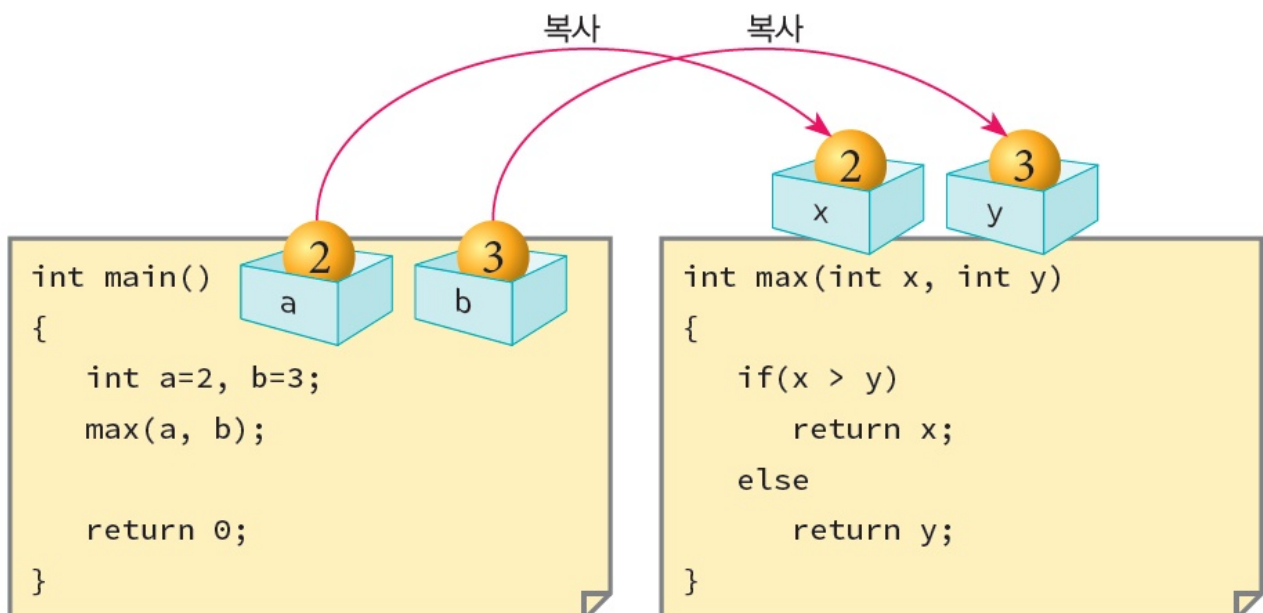
12

3.4 함수 호출 시 인수 전달 방법

- 값으로 호출하기(**call-by-value**): 호출하는 곳에서 인수를 전달할 때 인수의 값이 매개 변수로 복사되는 방법이다.
- 참조로 호출하기(**call-by-reference**): 원본 인수가 함수에 전달되는 방법이다. "참조로 호출하기" 방식에서 함수 안에서 매개 변수를 변경하면 원본 인수가 변경된다.

13

값으로 전달하기

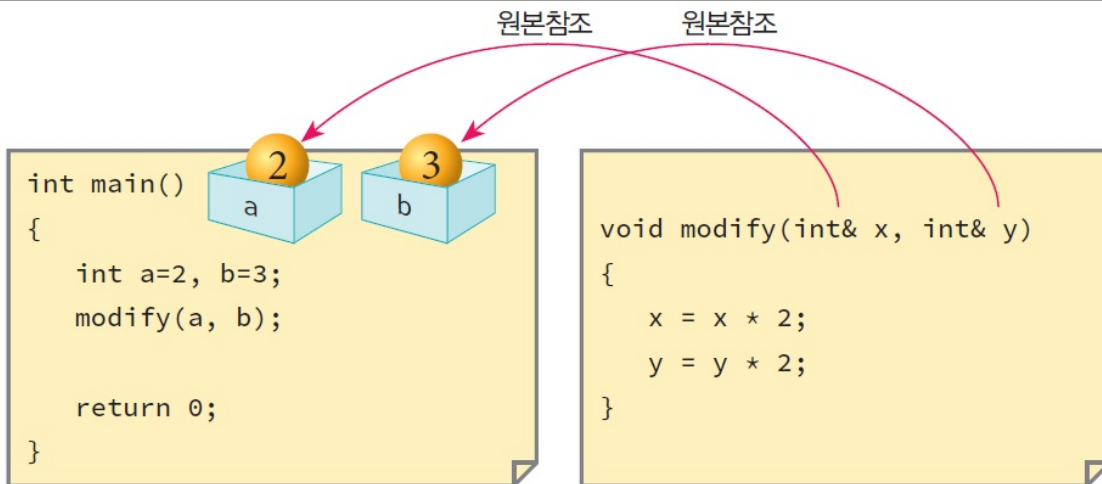


14

참조로 전달하기

- 참조자(reference)는 변수의 별명

```
int var = 10;  
int &ref = var;
```



15

Lab: swap() 함수 만들기

- `swap(a, b)`와 같이 호출하면 변수 `a`와 변수 `b`의 값을 교환

```
int main()  
{  
    int a = 100, b = 200;  
  
    printf("a=%d b=%d\n", a, b);  
    swap(a, b);  
    printf("a=%d b=%d\n", a, b);  
    return 0;  
}
```

16

Solution:

```
#include <stdio.h>
void swap(int& x, int& y)
{
    int tmp;
    tmp = x;
    x = y;
    y = tmp;
}
int main()
{
    int a = 100, b = 200;
    printf("a=%d b=%d\n", a, b);
    swap(a, b);
    printf("a=%d b=%d\n", a, b);
    return 0;
}
```

17

3.5 중복함수 (overloading)

- 동일한 이름의 함수를 여러 개 정의하는 것을 **중복 함수 (overloaded functions)**라고 한다.

```
// 정수값을 제공하는 함수
int square(int i)
{
    return i*i;
}

// 실수값을 제공하는 함수
double square(double i)
{
    return i*i;
}
```

18

중복 함수

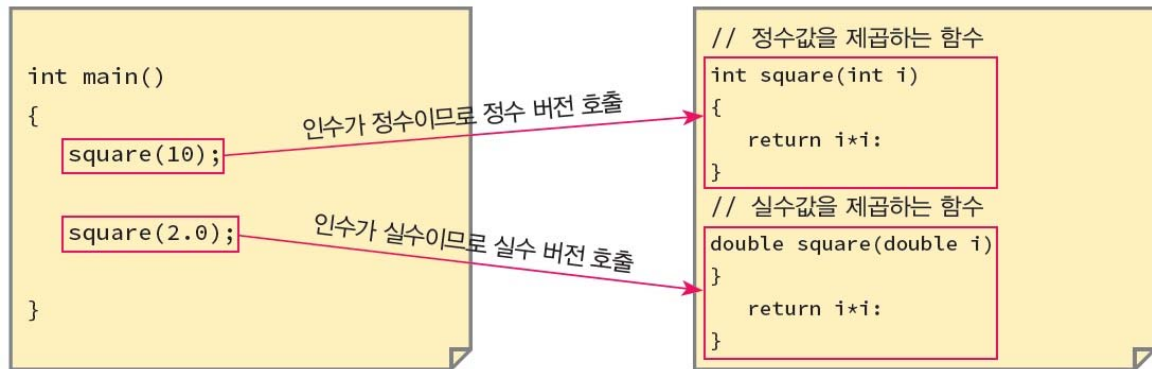


그림 3.3 중복 함수의 개념

19


예제

```
#include <iostream>
using namespace std;
int square(int i)
{
    cout << "square(int) 호출" << endl;
    return i*i;
}
double square(double i)
{
    cout << "square(double) 호출" << endl;
    return i*i;
}
int main()
{
    cout << square(10) << endl;
    cout << square(2.0) << endl;
    return 0;
}
```

20

Lab: 중복 함수

- 정수, 실수, 문자를 모두 출력할 수 있는 `print()` 함수를 중복 함수로 정의하고 사용해 보자.



```
C:\Windows\system32\cmd.exe
정수 출력: 100
실수 출력: 3.14
문자 출력: C
계속하려면 아무 키나 누르십시오 . . .
```

21

예제

```
#include <iostream>
using namespace std;
void print(int i) {
    cout << "정수 출력: " << i << endl;
}
void print(double f) {
    cout << "실수 출력: " << f << endl;
}
void print(char c) {
    cout << "문자 출력: " << c << endl;
}
int main()
{
    print(100);      // 정수를 출력하기 위하여 호출한다.
    print(3.14);     // 실수를 출력하기 위하여 호출한다.
    print('C');      // 문자를 출력하기 위하여 호출한다.
    return 0;
}
```

22

3.6 디폴트(default) 인수

- 인수를 전달하지 않아도 디폴트값을 대신 넣어주는 기능을 디폴트 인수 (default argument)라고 한다.

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

// 문자 c를 n번 반복하여 화면에 출력한다.
void display(char c = '*', int n = 10)
{
    for (int i = 0; i < n; i++)
        cout << c;
    cout << endl;
}
```

23

```
#include <iostream>
using namespace std;
void display(char c = '*', int n = 10)
{
    for (int i = 0; i < n; i++)
        cout << c;
    cout << endl;
}
int main()
{
    cout << "아무런 인수가 전달되지 않는 경우:\n";
    display();


    cout << "\n첫 번째 인수만 전달되는 경우:\n";
    display('#');

    cout << "\n모든 인수가 전달되는 경우:\n";
    display('#', 5);

    return 0;
}
```

24

실행결과

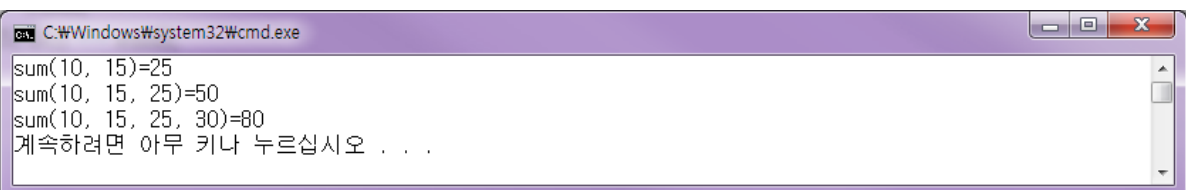


```
C:\Windows\system32\cmd.exe
아무런 인수가 전달되지 않는 경우:
*****
첫 번째 인수만 전달되는 경우:
#####
모든 인수가 전달되는 경우:
#####
계속하려면 아무 키나 누르십시오 . . .
```

25

Lab: 디폴트 매개 변수 실습하기

```
int main()
{
    cout << "sum(10, 15)=" << sum(10, 15) << endl;
    cout << "sum(10, 15, 25)=" << sum(10, 15, 25) << endl;
    cout << "sum(10, 15, 25, 30)=" << sum(10, 15, 25, 30) << endl;
    return 0;
}
```



```
C:\Windows\system32\cmd.exe
sum(10, 15)=25
sum(10, 15, 25)=50
sum(10, 15, 25, 30)=80
계속하려면 아무 키나 누르십시오 . . .
```

26

예제

```
#include <iostream>
using namespace std;
int sum(int x, int y, int z = 0, int w = 0)
{
    return x + y + z + w;
}

int main()
{
    cout << "sum(10, 15)=" << sum(10, 15) << endl;
    cout << "sum(10, 15, 25)=" << sum(10, 15, 25) << endl;
    cout << "sum(10, 15, 25, 30)=" << sum(10, 15, 25, 30) << endl;

    return 0;
}
```

27

3.7 인라인 함수

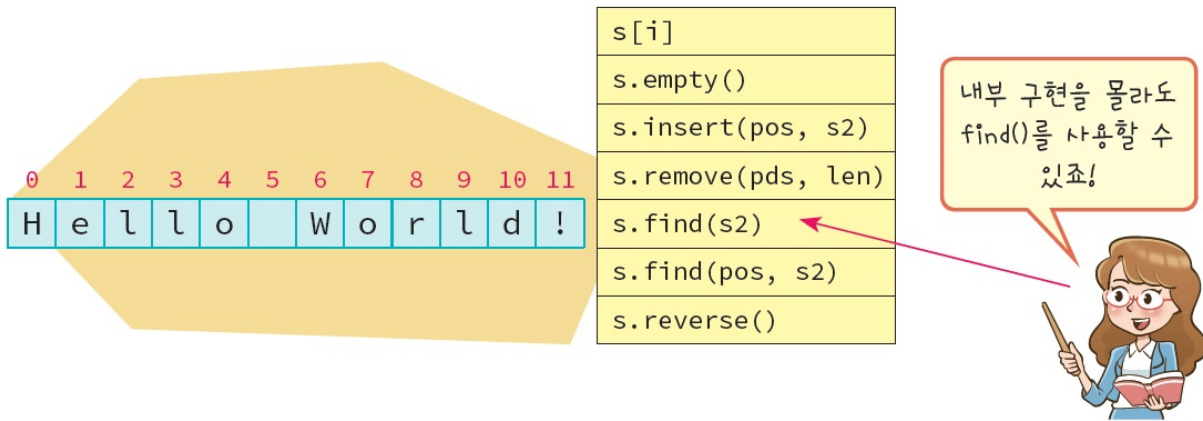
- 함수 이름 앞에 **inline**이 붙으면 컴파일러는 함수를 생성하지 않고 함수의 코드를 호출한 곳에 직접 집어넣는다.

```
// 실수값을 제공하는 함수
inline double square(double i)
{
    return i*i;
}
```

28

3.8 문자열

□ string 클래스



The diagram illustrates a string object in memory. On the left, a yellow hexagonal shape represents the string object, containing a sequence of characters: 'H', 'e', 'l', 'l', 'o', ' ', 'W', 'o', 'r', 'l', 'd', '!'. Each character is in a blue box, and the indices 0 through 11 are shown above them. To the right, a yellow box lists the methods available for the string object:

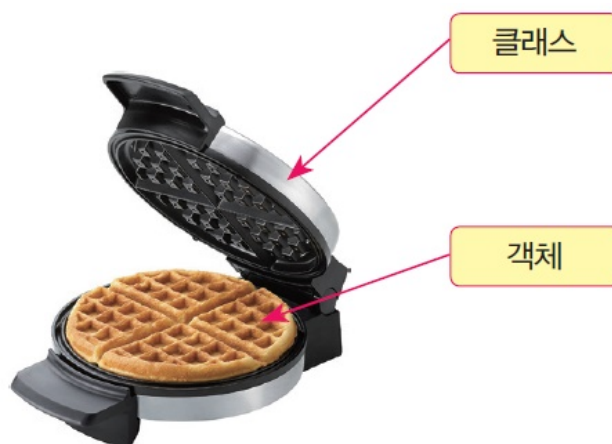
- s[i]
- s.empty()
- s.insert(pos, s2)
- s.remove(pos, len)
- s.find(s2)
- s.find(pos, s2)
- s.reverse()

A pink arrow points from the `s.find(s2)` method to a speech bubble that says: "내부 구현을 몰라도 find()를 사용할 수 있죠!" (Even if you don't know the internal implementation, you can use find()). A cartoon character of a girl with glasses and a blue jacket is pointing at the speech bubble.

29

string 객체

□ 클래스와 객체



30

string 클래스 사용하기

```
#include <string>
using namespace std;

void main()
{
    string s;                // string 객체 s를 생성한다.
    string s = "Hello World!"; // string 객체를 생성하고 초기화한다.
    string s{ "Hello World!" }; // string 객체를 생성하고 초기화한다.
}
```

문자열의 결합

```
#include <string>
using namespace std;

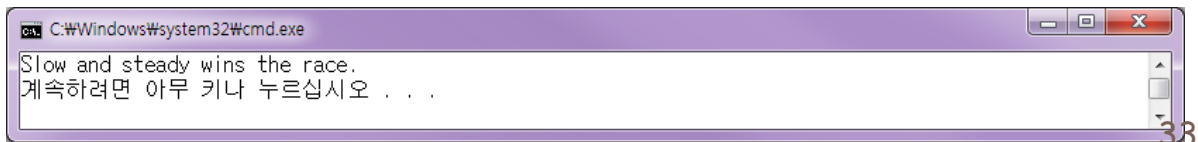
void main()
{
    string subject = "Money";
    string other = " has no value if it is not used";
    string sentence = subject + other;
}
```


예제

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

int main()
{
    string s1 = "Slow", s2 = "steady";
    string s3 = "the race.";
    string s4;

    s4 = s1 + " and " + s2 + " wins " + s3;
    cout << s4 << endl;
    return 0;
}
```



33

문자열의 비교

```
#include <string>
using namespace std;
void main()
{
    string s1 = "Hello", s2 = "World";
    if( s1 == s2 )
        cout << "동일한 문자열입니다." << endl;
    else
        cout << "동일한 문자열이 아닙니다." << endl;
    if( s1 > s2 )
        cout << "s1이 앞에 있습니다. " << endl;
    else
        cout << " s2가 앞에 있습니다. " << endl;
}
```

34

예제:

- 사용자로부터 이름과 주소를 받아서 친근하게 인사하는 프로그램을 작성



35

예제

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
    string s1, addr;
    cout << "이름을 입력하시오 : ";
    cin >> s1;
    cin.ignore(); // 엔터키를 없애기 위하여 필요하다.
    cout << "주소를 입력하시오 : ";
    getline(cin, addr);
    cout << addr << "의 " << s1 << "씨 안녕하세요? " << endl;
    return 0;
}
```

36

string 클래스 멤버 함수 사용

멤버 함수	설명
s[i]	i번째 원소
s.empty()	s가 비어있으면 true 반환
s.insert(pos, s2)	s의 pos 위치에 s2를 삽입
s.remove(pos, len)	s의 pos 위치에 len만큼을 삭제
s.find(s2)	s에서 문자열 s2가 발견되는 첫번째 인덱스를 반환
s.find(pos, s2)	s의 pos 위치부터 문자열 s2가 발견되는 첫번째 인덱스를 반환

37

예제

- 문자열 "When in Rome, do as the Romans."중에서 "Rome"이 몇 번째 위치에 있는지를 계산하는 프로그램



38

예제

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

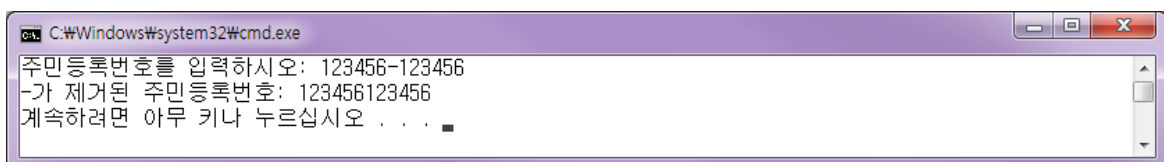
int main()
{
    string s="When in Rome, do as the Romans.";

    int index = s.find("Rome");
    cout << index << endl;
    return 0;
}
```

39

예제

- 사용자가 입력한 주민등록번호에서 '-' 문자를 삭제하는 프로그램을 작성하여 보자.



40

string 객체에서 문자 추출하기

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
    string s;
    cout << "주민등록번호를 입력하시오: ";
    cin >> s;

    cout << "-가 제거된 주민등록번호: ";
    for (auto& c : s) {
        if (c == '-') continue;
        cout << c;
    }
    cout << endl;
    return 0;
}
```

41

실습

- 사용자가 입력하는 전화번호에서 괄호 기호 (와)를 삭제한 형태로 출력하는 프로그램을 작성하라. 사용자가 “quit”를 입력하면 종료한다.

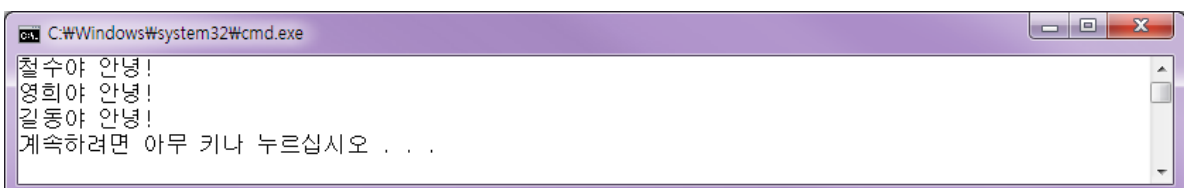
42

- 간단한 철자 교정 프로그램을 작성하여라. 문자열을 입력 받아서 문자열의 첫 번째 문자가 대문자인지를 검사한다. 만약 대문자가 아니면 대문자로 변환한다. 또한 문자열의 끝에 마침표가 존재하는지 검사한다. 역시 마침표가 없으면 넣어준다. 즉 입력된 문자열이 “c++ is easy”라면 “C++ is easy.”로 변환하여 화면에 출력한다.

문자열의 배열

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
    string list[] = { "철수", "영희", "길동" };

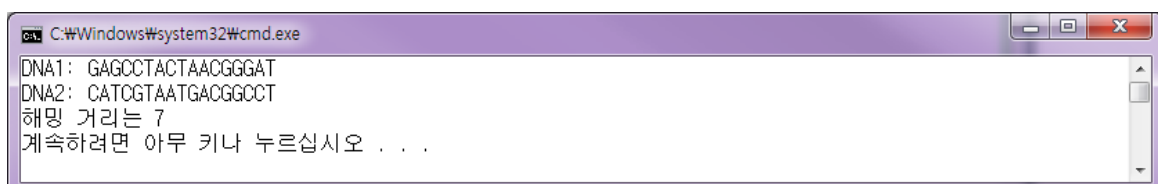
    for (auto& x : list)
        cout << (x + "야 안녕!") << endl;
    return 0;
}
```



- 사용자로부터 문자열 5개를 읽어서 가장 긴 문자열을 화면에 출력하라.

Lab: 해밍 거리 구하기

- 유전자를 나타내는 2개의 문자열을 받아서 동일한 위치에 틀린 글자가 몇 개나 있는지를 계산하는 프로그램을 작성해보자. 이것을 해밍 거리(Hamming distance)라고 한다



```
C:\Windows\system32\cmd.exe
DNA1: GAGCCTACTAACGGGAT
DNA2: CATCGTAATGACGGCCT
해밍 거리는 7
계속하려면 아무 키나 누르십시오 . . .
```

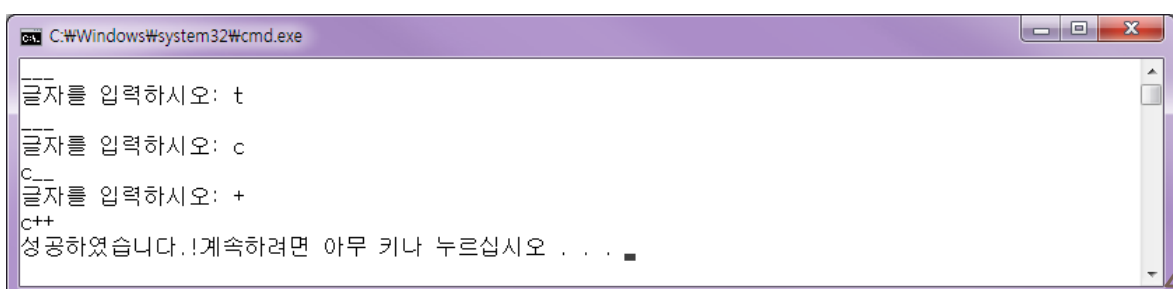
```

#include <iostream>
#include <string>
using namespace std;
int main() {
    string s1, s2;
    int count = 0;
    cout << "DNA1: ";
    cin >> s1;
    cout << "DNA2: ";
    cin >> s2;
    if (s1.length() != s2.length())
        cout << "오류: 길이가 다름" << endl;
    else {
        for (int i = 0; i < s1.length(); i++) {
            if (s1[i] != s2[i])
                count += 1;
        }
        cout << "해밍 거리는 " << count << endl;
    }
    return 0;
}

```

47

Lab: 행맨



48

solution

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
    char ch;
    string solution;
    string list[] =
    {
        "the",
        "c++",
        "programming",
        "language",
    };
};
```

49

```
int n = rand() % 4;
solution = list[n];
string guess(solution.length(), '_');
while (true) {
    cout << guess << endl;
    cout << "글자를 입력하시오: ";
    cin >> ch;
    for (int i=0; i< solution.length(); i++) {
        if (ch == solution[i]) {
            guess[i] = ch;
        }
    }
    if (solution == guess) {
        cout << solution << endl;
        cout << "성공하였습니다.!!";
        break;
    }
}
return 0;
}
```

50

실습

- 사용자로부터 암호를 입력받는다. 사용자의 암호가 해킹에 대하여 안전한지의 여부를 검사한다. 만약 암호 안에 대문자, 소문자, 숫자가 모두 들어있으면 안전한 암호로 간주한다. 만약 사용자의 암호가 3가지 종류의 문자를 다 가지고 있지 않으면 프로그램은 보안을 위하여 더 강한 암호를 고려하라고 제안한다.

실습

- 단어 애나그램(anagram) 게임을 작성해보자. 영어 단어를 이루는 글자들이 뒤죽박죽 섞인 것을 받아서 순서대로 재배치하는 게임을 애나그램 게임이라고 한다.
 - ▣ 문자열 안의 글자들을 섞으려면 난수가 필요하다. 2개의 난수를 발생시켜서 그 위치의 글자들을 서로 바꾸면 된다. 이것을 문자열의 길이만큼 반복한다. 물론 난수의 범위는 문자열 안이어야 한다.

Q & A

