




## Target: 번호 맞추기 게임



- 출제자가 낸 숫자를 예측하여 맞추는 게임
  - 값을 입력하면 정답과의 관계를 화면으로 출력해 줌
  - 점수를 계산하고 출력
  - 단, 출제자의 입력 숫자를 볼 수 없도록 숨겨야 함

```

C:\WINDOWS\system32\cmd.exe
출제자> 맞출 숫자를 입력하세요 (0~100) : **
[ 1회] 0 ~ 100 사이의 값 예측 =>60
더 큰 숫자입니다!
[ 2회] 60 ~ 100 사이의 값 예측 =>80
더 작은 숫자입니다!
[ 3회] 60 ~ 80 사이의 값 예측 =>70
더 큰 숫자입니다!
[ 4회] 70 ~ 80 사이의 값 예측 =>75
더 작은 숫자입니다!
[ 5회] 70 ~ 75 사이의 값 예측 =>73
성공 !!! 정답은 73
최종 점수 = 60
계속하려면 아무 키나 누르십시오 . . .
      
```

출제

예측

정답의 범위

게임 결과

2

## 2장 학습 목표



- 상수와 변수를 이해하고, 선언하고 활용하는 능력을 기른다.
- 자료형의 의미를 이해하고, 주어진 문제 해결을 위해 적절한 자료형을 선택할 수 있는 능력을 기른다.
- 연산자 우선순위와 결합 방향의 의미를 이해한다.
- 우선순위와 결합 방향에 따른 수식의 처리 순서를 이해한다.
- 부울식을 이해하고 활용하는 능력을 기른다.
- 여러 가지 분기문과 반복문을 정확히 이해하고 활용할 수 있는 능력을 기른다.

3

### 2.1 프로그램의 기본 요소



- 상수, 변수, 자료형
- 식별자
- 키워드

4

## 상수, 변수, 자료형



- 상수, 변수, 자료형



한번 만들면 변하지 않는 물건들



내용물이 변할 수 있는 물건들



다양한 유형(크기)의 커피 컵

5

## 식별자, 키워드



- 식별자(identifier):** 대상을 유일하게 구별할 수 있는 이름
  - 시작은 문자나 '\_', 대소문자 구별
  - a\_b c1 SIZE Length player // 적절한 식별자
  - 123 5a %rate size-1 gameover.cpp // 적절하지 않음(오류)
- 키워드(keyword):** 특별한 의미가 주어진 식별자

C++ Keywords						
asm	auto	bool	break	case	catch	char
class	const	const_cast	continue	default	delete	do
double	dynamic_cast	else	enum	explicit	export	extern
FALSE	float	for	friend	goto	if	inline
int	long	mutable	namespace	new	operator	private
protected	public	register	reinterpret_cast	return	short	signed
sizeof	static	static_cast	struct	switch	template	this
throw	true	try	typedef	typeid	typename	union
unsigned	using	virtual	void	volatile	wchar_t	while

6

## 2.2 상수, 변수, 자료형



- 변수
- 상수
- 자료형
  - Quiz?
- 문자 표현 방법
- Lab: 오버플로 발생 순간을 찾기

7

## 변수(variable)



### • 변수(variable)

```
char c; // 문자 변수 c 선언
int row, col; // int 변수 row와 col을 동시에 선언
int i = 7; // int 변수 i를 선언하고 7로 초기화
double interestRate=0.05; // double 변수 선언 및 초기화
```

변수 선언문	이름	공간(값)	자료형
<code>char ch;</code>	ch		char
<code>int year = 2017;</code>	year		int
<code>double rate = 0.05;</code>	rate		double

```
printf("year의 값=%d, 주소=%x\n", year, &year);
```

==> 실행결과 예: year의 값=2017, 주소=e6fc70

8

## 상수(constant)



- 원주율( $\pi$ )과 같이 변경될 수 없는 자료
- 기호 상수(symbolic constant) 또는 리터럴(literal)

```
#define PI 3.141592           // 전처리기 사용 상수 PI 선언
const double PI = 3.141592;  // const 키워드 사용
double area = PI*radius*radius; // 리터럴 PI의 사용 예
```

```
const double RateKphMph = 1.609344;
void main()
{
    int kph;
    double mph;
    printf("당신의 구속을 입력하십시오[Km/H]: ");
    scanf("%d", &kph);
    mph = kph / RateKphMph;
    printf("당신의 구속은 %1f [MPH] 입니다.\n", mph);
}
```

9

## 자료형(data type)

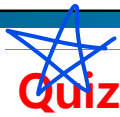


- C++ 기본 자료형
  - 정수형, 실수형, 부울형
  - bool 형 추가: true, false

자료형	용량 (bytes)	주요 용도	범위
정수형	char	1	문자(문자형), 또는 작은 정수 표현
	short	2	정수 표현
	int	4	큰 범위의 정수 표현
	long	4	큰 범위의 정수 표현
실수형	float	4	실수 표현
	double	8	유효 숫자가 많이 필요한 실수 표현 (float보다 두 배 정밀한 표현)
부울형	bool	1	참이나 거짓을 표현

→ x < 3

10



### • 다음의 자료형은?

3.141592 d  
12.3f f  
3 + 4 i  
'3' char  
3.0+4.0 d  
"3" c.p  
(long\_t) char x

11

## 문자 표현 방법



### • 문자 표현 방법

- 아스키코드(ASCII code)
  - 표준적인 8비트 문자 코드
  - 0에서 127까지의 숫자를 이용하여 문자를 표현함.
- 유니코드(unicode)
  - 표준적인 16비트 문자 코드
  - 전 세계의 모든 문자를 일관되게 표현하고 다룰 수 있도록 설계

### • 대입 연산자

- 변수의 값을 직접 변경하는 방법
- i = 7; // int형 변수 i에 7을 대입 (박스 i에 7을 넣음)
  - < 의 의미로 = 를 사용
  - = 의 의미로 == 를 사용

12

## Lab: 오버플로 발생 순간을 찾기



- short 자료형에서 오버플로가 발생하는 순간을 찾는 코드를 작성하시오.



- 조건
  - short 변수를 선언하고 1로 초기화
  - 변수를 1씩 증가하여 음수가 나오는 순간이 overflow
  - 변수 값을 같은 줄에 계속 출력하도록 함
  - 오버플로가 발생하면 다음 줄에 그때의 음수 값을 출력함
  - 오버플로 발생 순간 beep를 울리게 함
  - goto문 사용

13

## 구현



```
#include <stdio.h>
void main()
{
    short n = 0;
loop:
    n = n + 1;
    if (n > 0) {
        printf("\r short 최댓값 = %d", n);
        goto loop;
    }
    printf("\n오버플로우 발생\n");
    printf(" short 최솟값 = %d\n", n);
}
```

```
C:\WINDOWS\system32\cmd.exe
short 최댓값 = 32767
오버플로우 발생
short 최솟값 = -32768
계속하려면 아무 키나 누르십시오 . . .
```

14



- **고찰**

- short형에서 생각보다 크지 않은 값에서 오버플로우 발생
- 자료형의 선택이 중요함.
- 두 가지 이스케이프 시퀀스 'Wa'와 'Wr'을 활용
- int 형의 경우 ? 오버플로우 순간을 효과적으로 찾을 방법은?
- 실수형은?

15



## 2.3 수식과 연산자

- 수식(expression)
- 연산자의 분류
- 연산자에서 주의할 점
- 연산자 우선순위와 결합 방향
  - Quiz
- Lab. 섭씨 화씨 계산

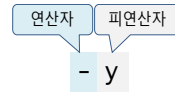
16



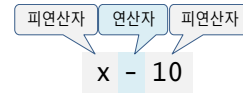
## 수식(expression)



- 수식(expression)
  - 상수, 변수, 연산자의 조합
  - 연산자와 피연산자



(a) 단항 연산자 -



(b) 이항 연산자 -

```
x + y
x*x + 5*x + 6
(principal * interest_rate * period) / 12.0
```

### 연산자의 분류

- 단항 연산자: 피연산자의 수가 1개 (ex:  $x++$ ,  $--y$ )
- 이항 연산자: 피연산자의 수가 2개 (ex:  $x+y$ )
- 삼항 연산자: 피연산자의 수가 3개 (ex:  $x > 0 ? X : -x$ )

17

## 연산자의 분류



연산의 종류에 따른 분류	연산자	의미
산술	$+ - * / \%$	더하기, 빼기, 곱하기, 나누기, 나머지(%)
비교	$> < == != >= <=$	관계 연산자: $< \Leftrightarrow >=$ , $== \Leftrightarrow !=$ , $> \Leftrightarrow <=$
증감	$++ --$	$a++$ , $++a$ , $b--$ , $--b$
논리	$\&\&    !$	연산 결과가 bool: true 또는 false
조건	$? :$	삼항 연산자. 예) $a = b ? c : d;$
비트 논리	$\&   \wedge \sim$	논리 연산을 비트별로 적용
비트 이동	$<< >>$	비트 단위로 좌/우로 밀어서 이동
대입	$=$ $+= -= *= /= \%=$ $>>= <<=  = \&=$	우측 연산항을 좌측 연산항으로 복사 좌측 연산항(변수)의 값이 바뀌게 됨 산술, 비트 논리, 비트 이동 연산을 포함 가능
기타	$( )$ $[ ]$ $.$ $->$	함수 호출 배열의 항목 추출 객체에서 내부의 항목 추출 포인터를 통한 객체 내부의 항목 추출
	$(type)$ $*$	형 변환 역참조 연산 (주소에서 값을 추출함)
	$\&$ $sizeof ( )$	주소 추출 연산 (변수의 주소 값을 추출) 자료형이나 변수의 크기 추출 연산
	$,$	콤마 연산

18

## 연산자에서 주의할 점



- 연산 결과의 자료형을 생각할 것:  $3+4$ ,  $3.0+4.0$  (int, double)
- 특히 정수의 나눗셈 연산에 주의
- 비교 연산의 결과는 bool형 → ex)  $3/4 \Rightarrow \text{int}=0$   
 $3.0/4 \Rightarrow \text{double}$
- 증감 연산자의 위치
- 단락 논리 (Short circuit logic)
- 조건 연산자  $a ? b : c$ ;
- 비트 검사 :  $(a \& (1 \ll (n-1))) == 0$
- 특별한 연산자
  - $()$ ,  $[]$ , (type)
  - sizeof, ,(comma),
  - $++$ ,  $--$
  - $\&$ ,  $*$

19

## 연산자 우선순위와 결합 방향



순위	연산자	결합 방향
1	$() [] -> . ++(\text{후위}) --(\text{후위})$	$->$ (좌에서 우)
2	sizeof &(주소) ++(전위) --(전위) ~ ! *(역참조) +(부호) -(부호), 형변환	$<-$ (우에서 좌)
3	$*(\text{곱셈}) / \%$	$->$ (좌에서 우)
4	$+(\text{덧셈}) -(\text{뺄셈})$	$->$ (좌에서 우)
5	$<< >>$	$->$ (좌에서 우)
6	$< <= >= >$	$->$ (좌에서 우)
7	$== !=$	$->$ (좌에서 우)
8	$\&(\text{비트연산})$	$->$ (좌에서 우)
9	$\wedge$	$->$ (좌에서 우)
10	$ $	$->$ (좌에서 우)
11	$\&\&$	$->$ (좌에서 우)
12	$  $	$->$ (좌에서 우)
13	$?(\text{삼항})$	$<-$ (우에서 좌)
14	$= += *= /= \%= \&= \wedge=  = <<= >>=$	$<-$ (우에서 좌)
15	$,(comma)$	$->$ (좌에서 우)

20

## • 우선순위 일반 지침

- 콤마 < 대입 < 논리 < 관계 < 산술 < 단항
- 괄호 연산자는 가장 우선순위가 높다.
- 모든 단항 연산자들은 이항 연산자들보다 우선순위가 높다.
- 콤마 연산자를 제외하고는 대입 연산자가 가장 우선순위가 낮다.
- 관계 연산자나 논리 연산자는 산술 연산자보다 우선순위가 낮다.

•  $x + 2 == y + 3$

자료형: bod

★ 연산자들의 우선 순위가 생각나지 않으면 괄호를 이용

$(x \leq 10) \&\& (y \geq 20)$

$(x < (y = (z = (k = 2))))$

## Quiz

- 다음 수식의 연산 순서를 괄호를 이용해 표시하라.

①  $a = (x + (3 * y))$

②  $a = ((5 / 9 * (b - 32.0)))$

③  $x = y = f = 'A';$

④  $c = (getchar() != 'Wn')$

- 다음과 같은 조건을 나타내는 조건식을 적어라.

① x는 0보다 작고 y는 0보다 크다.

$x < 0 \&\& y > 0$

② a가 5이상 10이하이거나 b가 20이다.

$(a \geq 5 \&\& a \leq 10) \parallel b == 20$

## Lab. 섭씨 화씨 계산

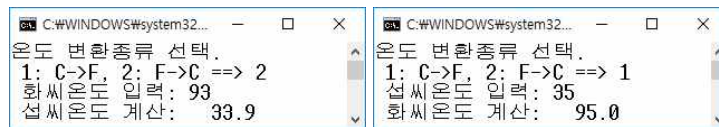


- 사용자로부터 섭씨를 화씨로 변환할 것인지 화씨를 섭씨로 변환할 것인지를 입력 받고, 서로 변환하여 결과를 출력하라.

– 섭씨온도(C)와 화씨온도(F)와의 관계식

$$C = (5/9)(F-32)$$

- 실행 결과



23

## 구현



```
void main()
{
    int    choice;
    double F, C;
    printf("온도 변환종류 선택.\n");
    printf(" 1: C->F, 2: F->C ==> ");
    scanf("%d", &choice);
    fflush(stdin);
    if( choice == 1 ) {    // 섭씨 -> 화씨
        printf(" 섭씨온도 입력: ");
        scanf( "%lf", &C );
        F = C * 9.0 / 5 + 32.;
        printf(" 화씨온도 계산: %6.1f\n", F);
    }
    if( choice == 2 ) {    // 화씨 -> 섭씨
        printf(" 화씨온도 입력: ");
        scanf( "%lf", &F );
        C = (F-32) * (5.0/9.0);
        printf(" 섭씨온도 계산: %6.1f\n", C);
    }
}
```

24

## 2.4 분기와 조건문



- 조건식(boolean expression)
- 분기문: if – else
- 분기문: switch – case – break - default
- Lab. 성적 입력, 학점 출력 프로그램
- 분기문: goto

25

## 조건식(boolean expression)



- 조건식 또는 부울식
  - 연산 결과가 참(true) 이나 거짓(false)이 되는 모든 식

```
(x < 10)           // x가 10보다 작으면 true 아니면 false
!(x < y)           // x가 y보다 작으면 false 아니면 true. →(x>=y)
(w <= 0) || (h >= 80) // w<=0이하 or h>=80이상 → true
(x > 3) && (x < 7)   // 3 < x < 7 이면 true 아니면 false
(3 < x < 7)         // 잘못된 조건식 식
```

- 반복문과 조건문에 들어가야 함
  - while (1 ) → while (true)
  - if( a+b ) → if (a+b != 0)

26

## 분기문: if - else



if 문	if ~ else 문
<pre>if (조건식) {     조건이 만족하면 실행되는 문장; } 항상 실행되는 문장;</pre>	<pre>if (조건식) {     조건이 만족하면 실행되는 문장; } else {     만족하지 않으면 실행되는 문장; } 항상 실행되는 문장;</pre>
else if 문	실제 의미
<pre>if ( 조건식1 ) {     조건식1이 만족하면 실행되는 문장; } else if ( 조건식2 ) {     조건식1이 만족하지 않고     조건식2가 만족해야 실행되는 문장; }</pre>	<pre>if ( 조건식1 ) {     문장1; } else {     if ( 조건식2 ) {         조건식1이 만족하지 않고         조건식2가 만족해야 실행되는 문장;     } }</pre>

27

## 분기문: switch – case – break - default



```
switch(expression){    // 정수 값 또는 수식을 계산하여
    case value1:        // expression 계산 결과가 value1 이면
        문장1;          // 문장1 수행
        break;          // switch 블록을 빠져 나감.
    case value2:        // expression 계산 결과가 value2 이면
        문장2;          // 문장2 수행
        break;
    ...
    default:            // 이상의 모든 경우가 아니면
        문장N;          // 문장N 수행
        break;
}
항상 실행되는 문장;
```

28

## Lab. 성적 입력, 학점 출력 프로그램

- 0에서 100점 사이의 점수를 입력 받아 다음 표와 같이 학점을 계산해 출력하는 프로그램을 구현하라.
  - if - else 사용
  - switch문 사용

점수	90점 이상	80~89	70~79	60~69	60점 미만
학점	A	B	C	D	F

29

## 분기문: goto

- 모듈화 된 프로그래밍을 방해
- 권장되지 않음.

```
goto a_label; // a_label로 이동하도록 한다.  
...  
a_label:      // 이것이 그 레이블이다.  
    처리할_문장들; // 이후 처리해야 할 문장들
```

30

## 2.5 반복문



- 반복문
- break와 continue
- 다중 반복문 빠져나오기
- Lab. Prime Number 프로그램
- Lab. 숫자 피라미드 만들기

31

## 반복문



- while 문

```
while ( 조건식 ) {           // 조건식이 true인 동안은
    statements;              // 블록을 반복함
}
```

- do-while 문

```
do {
    statements;              // 블록 실행 (최소 한번은 실행됨)
} while ( 조건식 );         // 조건식이 true인 동안 블록을 반복
```

- for문

```
for ( 초기값 설정부 ; 조건식 검사부 ; 증감식 처리부 ) {
    statements;
}
```

32



## break와 continue



- break 문

```
while ( 조건식 ) {  
    statements_1;  
    if ( 반복종료조건식 )    // 블록 내에서 어떤 조건이 되어  
        break;            // break 문을 만나면 빠져나와  
    statements_2;  
}  
statements_3;                // 다음 문장으로 간다.
```

- continue

```
while ( 조건식 ) {  
    statements_1;  
    if ( 반복계속조건식 )    // 조건이 되어 continue를 만나면  
        continue;          // 이후의 문장을 실행하지 않고  
    statements_2;            // 다시 while로 돌아가 조건식을 검사함  
}
```

33

## 다중 반복문 빠져나오기



- 다중 반복문 빠져나오기

```
for( int i=0 ; i<4 ; i++ ) {  
    for( int j=0 ; j<10 ; j++ ) {  
        printf("$");  
        if ( i==2 && j ==4 ) {    // 만약 i가 2이고 j가 4이면  
            i = 4;                // 조건식이 false가 되도록 처리하고  
            break;                // inner loop를 빠져 나옴  
        }                        // 그렇게 되면 이중 루프 전체를 빠져나오게 됨  
    }  
    printf("\n");  
}
```

```
$$$$$$$$$$  
$$$$$$$$$$  
$$$$$$
```

34

## Lab. Prime Number 프로그램



- 어떤 자연수  $n$ 을 입력 받아 2부터 그 수 사이에 있는 소수(prime number)를 모두 찾아 출력하고 소수의 개수도 함께 출력하는 프로그램을 구현해 보자.

```
C:\WINDOWS\system32\cmd.exe
소수를 구할 최대 숫자를 입력하십시오: 100
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53
59 61 67 71 73 79 83 89 97
2~100사이의 소수의 개수는 25개 입니다.
계속하려면 아무 키나 누르십시오 . . .
```

- 힌트

```
for( k=2; k<i ; k++) // 2부터 그 수보다 작은 수에 k로
    if(i%k == 0) break; // 나누어떨어지면 루프를 빠져나감
```

35



## Lab. 숫자 피라미드 만들기



- 높이를 입력받아 숫자 피라미드를 만들어보자.

```
C:\WINDOWS\system32\cmd.exe
높이를 입력하십시오: 7
      1
     1 2
    1 2 3
   1 2 3 4
  1 2 3 4 5
 1 2 3 4 5 6
1 2 3 4 5 6 7
계속하려면 아무 키나 누르십시오 . . .
```

- 힌트

```
for (int j=0 ; j<height-i ; j++)
    printf(" "); // 공백 출력
for (int j=0 ; j<i; j++)
    printf("%3d", j*2+1); // 증가하는 숫자 출력
for (int j=i-2; j>=0; j--)
    printf("%3d", j*2+1); // 감소하는 숫자 출력
```

36

## 2.6 응용: 근로 소득세 계산



- 공평한 근로 소득세
- 여러 가지 구현 방법

37

## 공평한 근로 소득세



- 소득을 입력하면 세금을 계산하고 세금과 세후 소득을 출력하는 프로그램을 작성하라. 수입이 조금이라도 많으면 세금을 세후 소득이 더 많아야 한다.

소득	근로소득세율
1200만원 이하	6%
1200만원 ~ 4600만원	15%
4600만원 ~ 8800만원	24%
8800만원 ~ 1억 5000만원	35%
1억 5000만원 초과	38%

38

## 여러 가지 구현 방법



```
printf("연봉을 입력하세요 ==> ");  
scanf("%d", &income);
```

```
if (income <= 1200) {  
    tax = income * 0.06;  
}  
if (1200 < income && income <= 4600) {  
    tax = 1200 * 0.06 + (income - 1200) * 0.15;  
}  
if (4600 < income && income <= 8800) {  
    tax = 1200 * 0.06 + (4600 - 1200) * 0.15 + (income - 4600) * 0.24;  
}  
if (8800 < income && income <= 15000) {  
    tax = 1200 * 0.06 + (4600 - 1200) * 0.15 + (8800 - 4600) * 0.24 + (income - 8800) * 0.35;  
}  
if (15000 < income) {  
    tax = 1200 * 0.06 + (4600 - 1200) * 0.15 + (8800 - 4600) * 0.24 + (15000 - 8800) * 0.35 + (income - 15000) * 0.38;  
}
```

```
printf("전체세금은 %.1f만원입니다.\n", tax);  
printf("순수소득은 %.1f만원입니다.\n", income - tax);
```

```
printf("연봉을 입력하세요 ==> ");  
scanf("%d", &income);  
in = income;
```

```
if (income > 15000) {  
    tax += (income - 15000) * 0.38;  
    income = 15000;  
}  
if (income > 8800) {  
    tax += (income - 8800) * 0.35;  
    income = 8800;  
}  
if (income > 4600) {  
    tax += (income - 4600) * 0.24;  
    income = 4600;  
}  
if (income > 1200) {  
    tax += (income - 1200) * 0.15;  
    income = 1200;  
}
```

```
tax += income * 0.06;  
printf("전체세금은 %.1f만원입니다.\n", tax);  
printf("순수소득은 %.1f만원입니다.\n", in - tax);
```

39

## 2.7 응용: 시큰둥한 게임



- Up And Down 게임
- 분석 및 설계
- 구현 및 결과

40

## Up And Down 게임

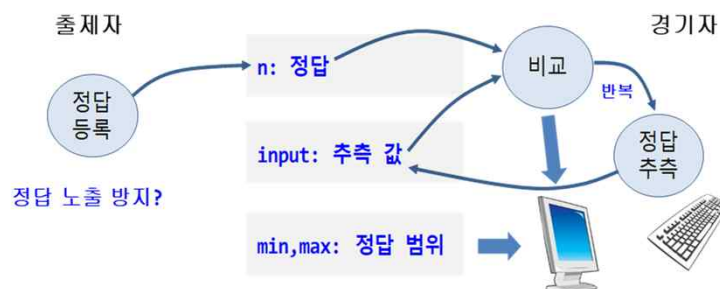


- 번호 맞추기 게임
  - 출제자가 컴퓨터에 두 자리의 숫자를 입력
  - 경기자가 이것을 추측하여 맞추는 게임
  - 예측한 숫자를 정답과 비교
    - "더 큰 숫자입니다" / "더 작은 숫자입니다" / "정답입니다" 출력
  - 중간에 맞히거나 10번 동안 맞히지 못하면 게임 종료
  - 점수:  $(10 - \text{추측횟수}) * 10\text{점}$
- 입력 조건
  - 출제자가 **입력한 숫자를** 경기자가 볼 수 없도록 해야 함
  - 힌트: `getch()` 함수 사용



41

## 분석 및 설계



42

## 구현 및 결과



```
...  
printf("두 자리 수 입력(1~99): ");  
char a = getch();  
printf("*");  
char b = getch();  
printf("*\n\n");  
n = (a-'0') * 10 + (b-'0');  
...
```

C:\WINDOWS\system32\cmd.exe  
출제자> 맞춰 숫자를 입력하세요 (0~100) : \*\*  
[ 1회] 0 ~ 100 사이의 값 예측 =>60  
더 큰 숫자입니다!  
[ 2회] 60 ~ 100 사이의 값 예측 =>80  
더 작은 숫자입니다!  
[ 3회] 60 ~ 80 사이의 값 예측 =>70  
더 큰 숫자입니다!  
[ 4회] 70 ~ 80 사이의 값 예측 =>75  
더 작은 숫자입니다!  
[ 5회] 70 ~ 75 사이의 값 예측 =>73  
성공 !!! 정답은 73  
최종 점수 = 60  
계속하려면 아무 키나 누르십시오 . . .

43

## 2장 요약문제, 연습문제, 실습문제



44



감사합니다!