객체지향프로그래밍 - 과제6

2019110634 컴퓨터공학과 이창렬

A. Code explanation & output analysis (Write the source code and results)

A-1. Listing 8.4

```
#include <iostream>
#include <cmath>
int main() {
  // Location of orbiting point is (x,y)
  double x; // These values change as the
  double y; // satellite moves
  const double PI = 3.14159;
  // Location of fixed point is always (100, 0),
  // AKA (p_x, p_y). Change these as necessary.
  const double p_x = 100;
  const double p_y = 0;
  // Radians in 10 degrees
  const double radians = 10 * PI / 180;
  // Precompute the cosine and sine of 10 degrees
  const double COS10 = cos(radians);
  const double SIN10 = sin(radians);
  // Get starting point from user
  std::cout << "Enter initial satellite coordinates (x,y):";
  std::cin >> x >> y;
  // Compute the initial distance
  double d1 = sqrt((p_x - x) * (p_x - x) + (p_y - y) * (p_y - y));
  // Let the satellite orbit 10 degrees
  double x_old = x; // Remember x's original value
  x = x * COS10 - y * SIN10; // Compute new x value
  // x's value has changed, but y's calculate depends on
  // x's original value, so use x_old instead of x.
  y = x_old * SIN10 + y * COS10;
  // Compute the new distance
  double d2 = sqrt((p_x - x) * (p_x - x) + (p_y - y) * (p_y - y));
  // Print the difference in the distances
  std::cout << "Difference in distances: " << d2 - d1 << '\n';
}
```

```
Enter initial satellite coordinates (x,y):1201204 49231.13123
Difference in distances: 2.22923
```

이 코드는 (100,0)이라는 기준 좌표를 설정하고 사용자의 좌표를 입력받은 뒤 cmath의 제곱근 함수를 사용해 거리를 구하고 사용자 좌표가 10도 회전했을때의 거리를 구해서 그 둘

의 차를 나타내는 것입니다.

A-2. Listing 8.12

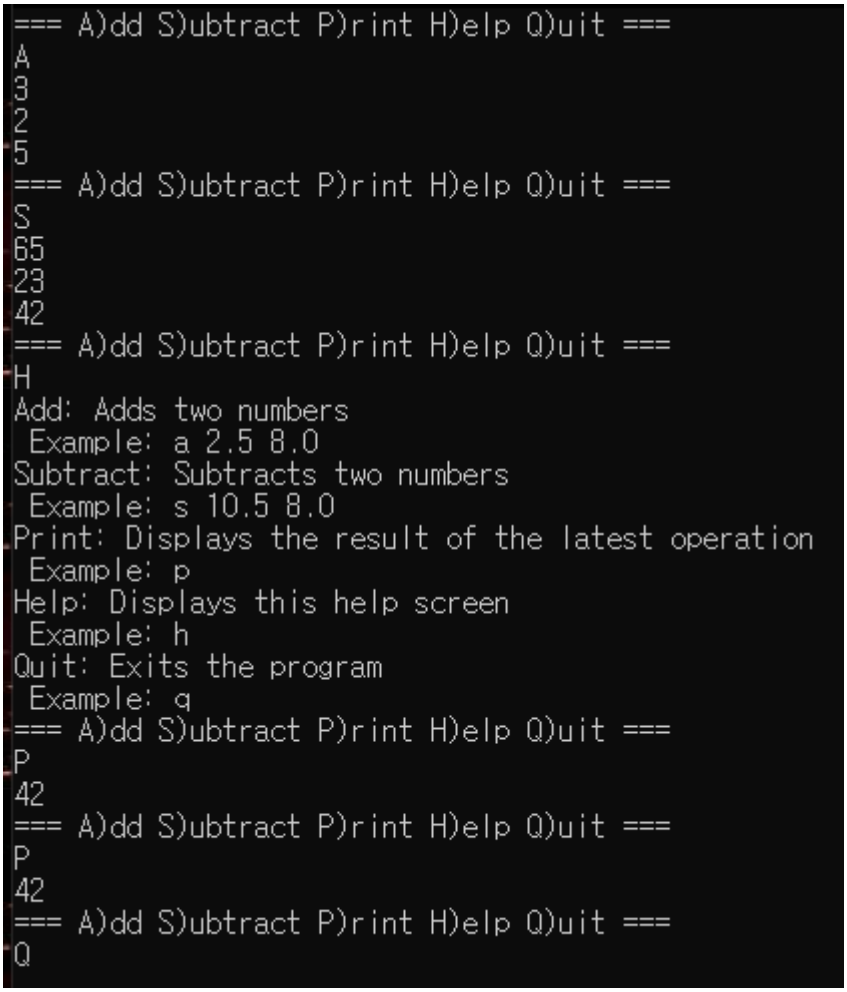
```
#include <iostream>
#include <cstdlib>
#include <ctime>
int main() {
  // Set the random seed value
  srand(static_cast<unsigned>(time(0)));
  // Roll the die three times
  for (int i = 0; i < 3; i++) {
    // Generate random number in the range 1...6
    int value = rand() % 6 + 1;
    // Show the die
    std::cout << "+-------+\n";
    switch (value) {
    case 1:
      std::cout << "| |\n";
      std::cout << "| * |\n";
      std::cout << "| |\n";
      break;
    case 2:
      std::cout << "| * |\n";
      std::cout << "| |\n";
      std::cout << "| * |\n";
      break;
    case 3:
      std::cout << "| * |\n";
      std::cout << "| * |\n";
      std::cout << "| * |\n";
      break;
    case 4:
      std::cout << "| * * |\n";
      std::cout << "| |\n";
      std::cout << "| * * |\n";
      break;
    case 5:
      std::cout << "| * * |\n";
      std::cout << "| * |\n";
      std::cout << "| * * |\n";
      break;
    case 6:
      std::cout << "| * * * |\n";
      std::cout << "| |\n";
      std::cout << "| * * * |\n";
      break;
    default:
      std::cout << " *** Error: illegal die value ***\n";
      break;
    }
    std::cout << "+-------+\n";
  }
}
```

프로그램을 시작한 시간에 따라 seed값으로 srand를 정하고 value값에 1~6까지 중 하나로 나오는 것으로 하고 이를 3번 반복합니다.

A-3. Listing 9.11

```
#include <iostream>
#include <cmath>
/*
* help_screen
* Displays information about how the program works
* Accepts no parameters
* Returns nothing
*/
void help_screen() {
  std::cout << "Add: Adds two numbers\n";
  std::cout << " Example: a 2.5 8.0\n";
  std::cout << "Subtract: Subtracts two numbers\n";
  std::cout << " Example: s 10.5 8.0\n";
  std::cout << "Print: Displays the result of the latest operation\n";
  std::cout << " Example: p\n";
  std::cout << "Help: Displays this help screen\n";
  std::cout << " Example: h\n";
  std::cout << "Quit: Exits the program\n";
  std::cout << " Example: q\n";
}
/*
* menu
* Display a menu
* Accepts no parameters
* Returns the character entered by the user.
*/
char menu() {
  // Display a menu
  std::cout << "=== A)dd S)ubtract P)rint H)elp Q)uit ===\n";
```

```
  // Return the char entered by user
  char ch;
  std::cin >> ch;
  return ch;
}
/*
* main
* Runs a command loop that allows users to
* perform simple arithmetic.
*/
int main() {
  double result = 0.0, arg1, arg2;
  bool done = false; // Initially not done
  do {
    switch (menu()) {
    case 'A': // Addition
    case 'a':
      std::cin >> arg1 >> arg2;
      result = arg1 + arg2;
      std::cout << result << '\n';
      break;
    case 'S': // Subtraction
    case 's':
      std::cin >> arg1 >> arg2;
      result = arg1 - arg2;
      // Fall through, so it prints the result
    case 'P': // Print result
    case 'p':
      std::cout << result << '\n';
      break;
    case 'H': // Display help screen
    case 'h':
      help_screen();
      break;
    case 'Q': // Quit the program
    case 'q':
      done = true;
      break;
    }
  } while (!done);
}
```



먼저 'A'나 'a'가 입력되면 사용자가 입력한 두 수를 더한 결과가 나오고

'B'나 'b'가 입력되면 사용자가 입력한 두 수의 차가 나오며

'P'나 'p'를 입력하면 최근의 결과값이 나옵니다.

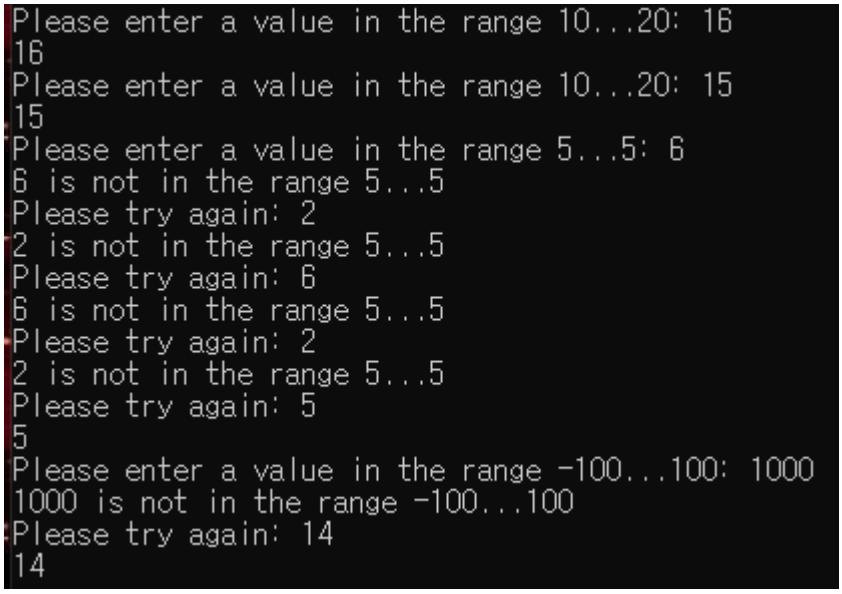
'H'나 'h'를 입력하면 help_screen의 예시들이 출력되고 'Q'나 'q'가 입력되면 프로그램이 종료됩니다.

가장자리의 do while문은 'Q'나 'q'의 case문의 반대로 입력하여 프로그램이 종료되지 않는 한 무한으로 실행되게 합니다.

A-4. Listing 9.12

```
#include <iostream>
/*
* get_int_range(first, last)
* Forces the user to enter an integer within a
* specified range
* first is either a minimum or maximum acceptable value
* last is the corresponding other end of the range,
```

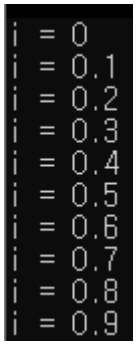
```
* either a maximum or minimum * value
* Returns an acceptable value from the user
*/
int get_int_range(int first, int last) {
  // If the larger number is provided first,
  // switch the parameters
  if (first > last) {
    int temp = first;
    first = last;
    last = temp;
  }
  // Insist on values in the range first...last
  std::cout << "Please enter a value in the range "
    << first << "..." << last << ": ";
  int in_value; // User input value
  bool bad_entry;
  do {
    std::cin >> in_value;
    bad_entry = (in_value < first || in_value > last);
    if (bad_entry) {
      std::cout << in_value << " is not in the range "
        << first << "..." << last << '\n';
      std::cout << "Please try again: ";
    }
  } while (bad_entry);
  // in_value at this point is guaranteed to be within range
  return in_value;
}
/*
* main
* Tests the get_int_range function
*/
int main() {
  std::cout << get_int_range(10, 20) << '\n';
  std::cout << get_int_range(20, 10) << '\n';
  std::cout << get_int_range(5, 5) << '\n';
  std::cout << get_int_range(-100, 100) << '\n';
}
```



각 범위 내의 숫자를 입력하지 않으면 다시 입력하라는 말이 나오고 제대로 된 숫자를 입력하면 입력한 숫자가 무엇인지 나옵니다.

A-5. Listing 9.17

```
#include <iostream>
#include <cmath>
/*
* equals(a, b, tolerance)
* Returns true if a = b or |a - b| < tolerance.
* If a and b differ by only a small amount
* (specified by tolerance), a and b are considered
* "equal." Useful to account for floating-point
* round-off error.
* The == operator is checked first since some special
* floating-point values such as HUGE_VAL require an
* exact equality check.
*/
bool equals(double a, double b, double tolerance) {
  return a == b || fabs(a - b) < tolerance;
}
int main() {
  for (double i = 0.0; !equals(i, 1.0, 0.0001); i += 0.1)
    std::cout << "i = " << i << '\n';
}
```

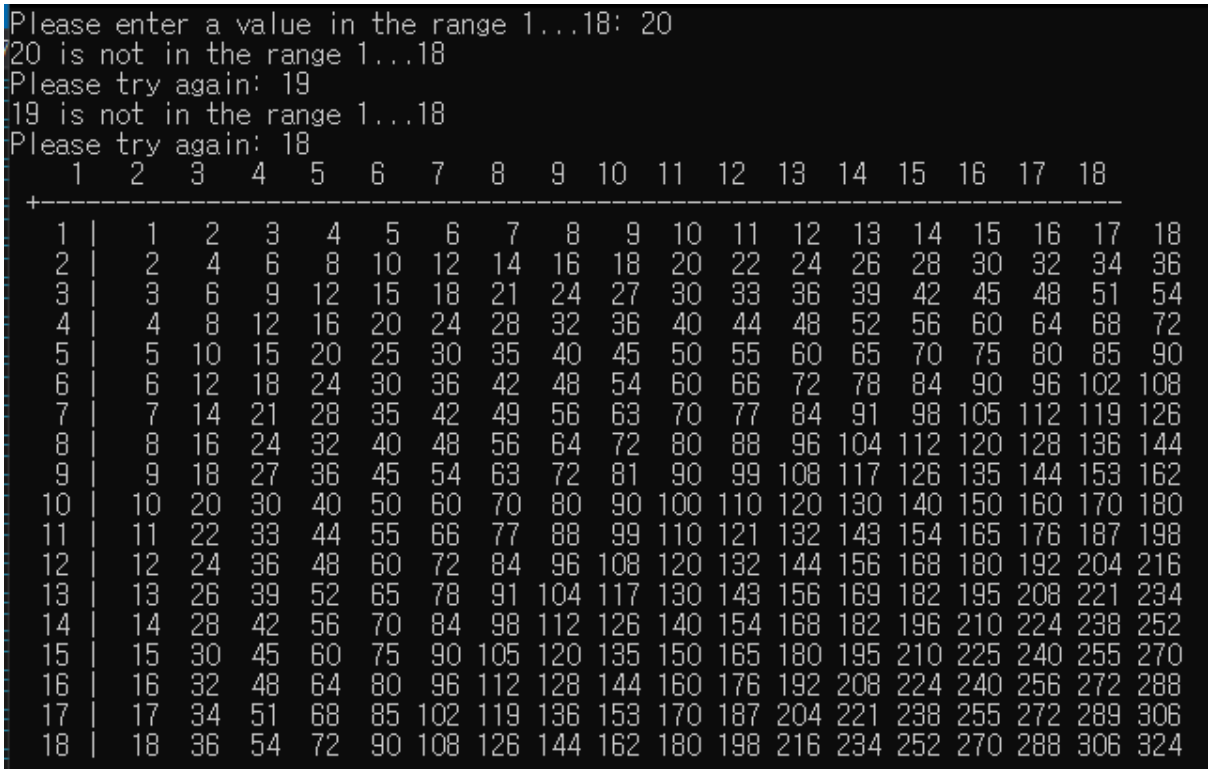


a엔 변수를 b엔 1.0을 두고서 숫자가 다르면 a에 0부터 0.1씩 추가해서 출력을 하고 a라는 변수와 b가 같으면 프로그램을 종료합니다.

A-6. Listing 9.18

```
#include <iostream>
#include <iomanip>
// Print the column labels for an n x n multiplication table.
void col_numbers(int n) {
  std::cout << " ";
  for (int column = 1; column <= n; column++)
    std::cout << std::setw(4) << column; // Print heading for this column.
  std::cout << '\n';
}
// Print the table's horizontal line at the top of the table
// beneath the column labels.
void col_line(int n) {
  std::cout << " +";
  for (int column = 1; column <= n; column++)
    std::cout << "----"; // Print separator for this row.
  std::cout << '\n';
}
// Print the title of each column across the top of the table
// including the line separator.
void col_header(int n) {
  // Print column titles
  col_numbers(n);
  // Print line separator
  col_line(n);
}
// Print the title that appears before each row of the table's
// body.
void row_header(int n) {
  std::cout << std::setw(4) << n << " |"; // Print row label.
}
// Print the line of text for row n
// This includes the row number and the
// contents of each row.
void print_row(int row, int columns) {
  row_header(row);
  for (int col = 1; col <= columns; col++)
    std::cout << std::setw(4) << row * col; // Display product
```

```
  std::cout << '\n'; // Move cursor to next row
}
// Print the body of the n x n multiplication table
void print_contents(int n) {
  for (int current_row = 1; current_row <= n; current_row++)
    print_row(current_row, n);
}
// Print a multiplication table of size n x n.
void timestable(int n) {
  // First, print column heading
  col_header(n);
  // Print table contents
  print_contents(n);
}
// Forces the user to enter an integer within a
// specified range first is either a minimum or maximum
// acceptable value last is the corresponding other end
// of the range, either a maximum or minimum value
// Returns an acceptable value from the user
int get_int_range(int first, int last) {
  // If the larger number is provided first,
  // switch the parameters
  if (first > last) {
    int temp = first;
    first = last;
    last = temp;
  }
  // Insist on values in the range first...last
  std::cout << "Please enter a value in the range "
    << first << "..." << last << ": ";
  int in_value; // User input value
  bool bad_entry;
  do {
    std::cin >> in_value;
    bad_entry = (in_value < first || in_value > last);
    if (bad_entry) {
      std::cout << in_value << " is not in the range "
        << first << "..." << last << '\n';
      std::cout << "Please try again: ";
    }
  } while (bad_entry);
  // in_value at this point is guaranteed to be within range
  return in_value;
}
int main() {
  // Get table size from user; allow values in the
  // range 1...18.
  int size = get_int_range(1, 18);
  // Print a size x size multiplication table
  timestable(size);
}
```



1~18까지의 숫자가 아니면 다시 입력하라는 말이 나오고 올바른 값을 입력했을 때 그 값에 맞춰서 곱셈값이 나옵니다. 이때 print_contents의 print_row를 통해 for문을 돌려 곱셈값을 출력합니다.

B. Exercises(Write the questions down on your answer sheet) p.197 - 199

3. Which one of the following values could be computed by the rand function?

4.5 34 -1 RAND_MAX + 1

>> 34입니다.

8. Consider each of the following code fragments below that could be part of a C++ program. Each
fragment contains a call to a standard C/C++ library function. Answer each question in one of the
following three ways:

- If the code fragment contains a compile-time error, write the word *error* for the answer.

- If the code fragment contains no compile-time errors and you can determine its output at compile-time, provide the fragment's literal output.
- If the code fragment contains no compile-time errors but you cannot determine its exact output at compile-time, provide one possible evaluation and write the word *example* for the answer and provide one possible literal output that the code fragment could produce.

(a) std:: cout << sqrt(4.5) << '\n';

2.12132

(b) std:: cout << sqrt(4.5, 3.1) << '\n';

error, 해당 함수는 인수를 하나만 사용합니다.

(c) std:: cout << rand(4) << '\n';

error, rand함수는 인자를 받지 않습니다.

(d) double d = 16.0;

std:: cout << sqrt(d) << '\n';

4

(e) std:: cout << srand( ) << '\n';

error

(f) std:: cout << rand( ) << '\n';

41

(g) int i = 16;

std:: cout << sqrt(i) << '\n';

4

(h) std:: cout << srand(55) << '\n';

error

(i) std:: cout << tolower('A') << '\n';

97

(k) std:: cout << sqrt( ) << '\n';

error

(l) std:: cout << toupper('E') << '\n';

69

(m) std::cout << toupper('e') << '\n';

69

(n) std::cout << toupper("e") << '\n';

error

(o) std::cout << exp(4.5) << '\n';

90.0171

(p) std::cout << toupper('h', 5) << '\n';

error, toupper는 2개의 인수를 사용하지 않습니다.

(q) std::cout << ispunct('!') << '\n';

16

(r) std::cout << tolower("F") << '\n';

error

(s) char ch = 'D';

    std::cout << tolower(ch) << '\n';

100

(t) std::cout << exp(4.5, 3) << '\n';
error, exp는 인자를 하나만 받습니다.

(u) std::cout << toupper('7') << '\n';
55

(v) double a = 5, b = 3;

    std::cout << exp(a, b) << '\n';

error, exp는 인자를 하나만 받습니다.

(w) std::cout << exp(3, 5, 2) << '\n';

error, exp는 인자를 하나만 받습니다.

(x) std::cout << tolower(70) << '\n';
102

(y) double a = 5;

    std::cout << exp(a, 3) << '\n';

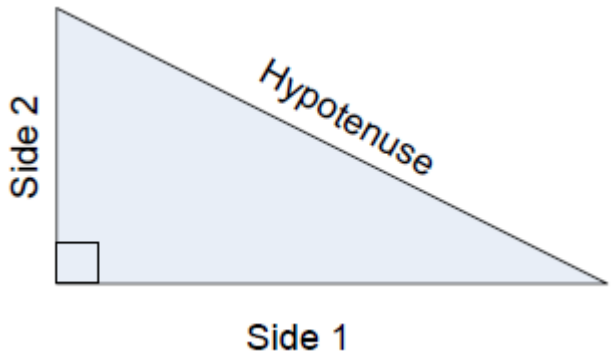
error, exp는 인자를 하나만 받습니다.

(z) double a = 5;

std::cout << exp(3, a) << '\n';

error, exp는 인자를 하나만 받습니다.

9. From geometry: Write a computer program that given the lengths of the two sides of a right triangle adjacent to the right angle computes the length of the hypotenuse of the triangle. (See Figure 8.6.) If you are unsure how to solve the problem mathematically, do a web search for the Pythagorean theorem.




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

int main() {
  int x, y, h = 0;
  cout << "x : ";
  cin >> x;
  cout << "y : ";
  cin >> y;

  h = sqrt(x * x + y * y);
  cout << "빗변의 길이 :" << h;
}
```

C. Exercises(Write the questions down on your answer sheet) p.236 - 240

1~8 : If the code does not work, then explain what is wrong, and correct the code

1. Is the following a legal C++ program?

```
int proc(int x) {
  return x + 2;
```

```
}
int proc(int n) {
  return 2 * n + 1;
}
int main() {
  int x = proc(5);
}
```

No. #include <iostream>이라는 헤더를 작성해야합니다. 또한 proc 함수가 두번 사용되었습니다.

2. Is the following a legal C++ program?

```
int proc(int x) {
  return x + 2;
}
int main() {
  int x = proc(5),
    y = proc(4);
}
```

No. #include <iostream>이라는 헤더를 작성해야합니다.

3. Is the following a legal C++ program?

```
#include <iostream>
void proc(int x) {
  std::cout << x + 2 << '\n';
}
int main() {
  int x = proc(5);
}
```

No. x는 int값이고 proc는 void이기 때문입니다.

4. Is the following a legal C++ program?

```
#include <iostream>
void proc(int x) {
  std::cout << x + 2 << '\n';
}
int main() {
```

```
  proc(5);
}
```

Yes.

5. Is the following a legal C++ program?

```
#include <iostream>
int proc(int x, int y) {
  return 2 * x + y * y;
}
int main() {
  std::cout << proc(5, 4) << '\n';
}
```

Yes.

6. Is the following a legal C++ program?

```
#include <iostream>
int proc(int x, int y) {
  return 2 * x + y * y;
}
int main() {
  std::cout << proc(5) << '\n';
}
```

No. proc의 parameter는 두개인데 main에서 proc에 하나만 입력했기 때문입니다.

7. Is the following a legal C++ program?

```
#include <iostream>
int proc(int x) {
  return 2 * x * x;
}
int main() {
  std::cout << proc(5, 4) << '\n';
}
```

No. proc의 parameter는 한 개이지만 main에서는 두개를 입력했습니다.

8. Is the following a legal C++ program?

```
#include <iostream>
proc(int x) {
  std::cout << 2 * x * x << '\n';
}
int main() {
  proc(5);
}
```

No. proc의 변수타입이 지정되지 않았습니다.

9. The programmer was expecting the following program to print 200. What does it print instead? Why does it print what it does?

```
#include <iostream>
void proc(int x) {
  x = 2 * x * x;
}
int main() {
  int num = 10;
  proc(num);
  std::cout << num << '\n';
}
```

10

proc에서 return으로 전달한 것이 아닌 main에 있던 10이 출력됐습니다.

10. Is the following program legal since the variable x is used in two different places (proc and main)? Why or why not?

```
#include <iostream>
int proc(int x) {
  return 2 * x * x;
}
int main() {
  int x = 10;
  std::cout << proc(x) << '\n';
}
```

main에서의 x와 proc에서의 x는 아무 상관이 없기 때문에 정당하게 쓰였습니다.

D. Additional exercises

D-1. Write a program that reads a series of numbers and calculates the average, geometric mean, and harmonic mean.

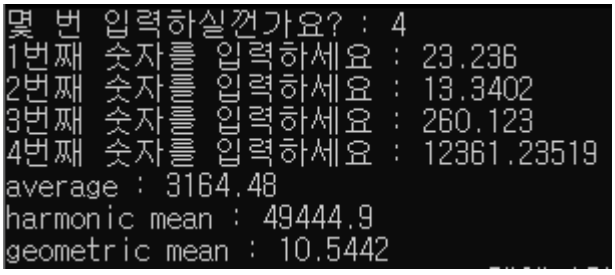
```
#include <iostream>
#include <cmath>
using namespace std;

float harmonicMean(double a) {
  float sum = 0;
  sum += 1 / a;
  return sum;
}

float geometricMean(double a) {
  float sum = 1;
  sum *= a;
  return sum;
}

int main() {
  double sum = 0, average = 0, geometric_mean = 0, harmonic_mean = 0;
  double count = 0, a;
  cout << "몇 번 입력하실껀가요? : ";
  cin >> count;

  for (int i = 0; i < count; i++) {
    cout << i + 1 << "번째 숫자를 입력하세요 : ";
    cin >> a;
    sum += a;
    harmonic_mean = harmonicMean(a);
    geometric_mean = geometricMean(a);
  }
  harmonic_mean = count / harmonic_mean;
  geometric_mean = pow(geometric_mean, 1.0 / count);
  average = sum / count;
  cout << "average : " << average << endl;
  cout << "harmonic mean : " << harmonic_mean << endl;
  cout << "geometric mean : " << geometric_mean;
}
```



D-2. Write a program to print Fibonacci series (0, 1, 1, ..., 34)

Use the user-defined function: int Fibonacci (int n)

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

int main() {
    int n, a = 0, b = 1, nextTerm = 0;

    cout << "Enter the number of terms: ";
    cin >> n;

    cout << "Fibonacci Series: ";

    for (int i = 1; i <= n; ++i) {
        // Prints the first two terms.
        if(i == 1) {
            cout << a << ", ";
            continue;
        }
        if(i == 2) {
            cout << b << ", ";
            continue;
        }
        nextTerm = a + b;
        a = b;
        b = nextTerm;

        cout << nextTerm << ", ";
    }
    return 0;
}
```

D-3. Write a program that calculates the real solution of the quadratic equation ax^2 + bx + c = 0

- Read in the values for the parameters a, b, c (type double)
- Then the program should calculate the solution considering the following circumstances:
- a=0 and b=0 → Not a valid equation

- a=0 and b≠0 → x= -c/b
- b^2 - 4ac < 0 → Not a Real Solution
- b^2 - 4ac
  >=0 →

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

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

int main() {
  double a, b, c, x1, x2;
  char x;
  cout << "이차항의 계수는 : ";
  cin >> a;
  cout << "일차항의 계수는 : ";
  cin >> b;
  cout << "상수항의 계수는 : ";
  cin >> c;
  cout << "입력하신 식은 다음과 같습니다.";
  cout << a << "x^2 + " << b << "x + " << c << endl;
  if (a == 0 && b == 0){
    cout << "not a valid equation";
  }
  if (a == 0 && b != 0) {
    x1 = -(c / b);
    cout << "root = " << x1 << endl;
  }
  if ((b * b - 4 * a * c) > 0) {
    x1 = (-b + sqrt(b * b - 4 * a * c)) / 2 * a;
    x2 = (-b - sqrt(b * b - 4 * a * c)) / 2 * a;
    cout << "x = " << x1 << " ," << x2 << endl;
  }
  else if ((b * b - 4 * a * c) < 0) {
    cout << "not a real solution" << endl;
  }
}
```

```
이차항의 계수는 : 1
일차항의 계수는 : -3
상수항의 계수는 : 2
입력하신 식은 다음과 같습니다.1x^2 + -3x + 2
x = 2 ,1
```

* Wrtie and test the following functions

D-4. A function that returns the permutation of n and r.

```
void permutation(int depth){
    if(depth == r){
        printArray(pArr);
        return;
    }

    for(int i = 1; i <= n; i++){
        if(!check[i]){
            check[i] = true;
            pArr[depth] = i;
            permutation(depth + 1);
            check[i] = false;
        }
    }
}

int main(){
  permutation(0);
}
```