Object-Oriented Programming Lab #05

Department: 응용물리학과 Student ID: 2017103038 Name: 권인회

A. Exercises (Write the questions down on your answer sheet)

(pp. 153-157), Exercises 1, 3, 8, 12, 14, 15, 19, 22
(write output analysis for all exercises)

1. In Listing 6.4 (addnonnegatives.cpp) could the condition of the if statement have used > instead of >= and achieved the same results? Why?

: 같은 결과가 나온다. 변수 input에 0의 값이 들어갔을 때에서의 차이인데, if statement가 >= 일 때는 sum += input 코드가 처리가 된다. 하지만 input은 0이기 때문에 sum에 input을 더해도 값이 변하지 않는다. 마찬가지로, if statement가 > 일때는 sum += input 코드가 처리되지 않기 때문에 sum의 값에 변화가 없다. 두 상황 다 sum의 값에 변화가 생기지 않으므로 if statement가 >= 대신에 > 를 사용해도 같은 결과가 나온다. 하지만, while statement에 적용하면 다른 결과가 나온다.

3. Use a loop to rewrite the following code fragment so that it uses just one std::cout and one '₩n'. std::cout << 2 << '₩n';

std::cout << 4 << '₩n';

std::cout << 6 << '₩n';

std::cout << 8 << '₩n';

std::cout << 10 << '₩n';

std::cout << 12 << '₩n';

std::cout << 14 << '₩n';

std::cout << 16 << '₩n';

: < 코드 >

```
#include <iostream>
int main() {
        int a = 2;
        while (a <= 16)
        {
```

```
                std::cout << a << '₩n';
                a += 2;
        }
}
```

< 실행결과 >

2

4

6

8

10

12

14

16

C:₩Users₩Administrator₩source₩repos₩Ex002₩Debug₩Ex002.exe(프로세스 37816개)이(가

) 종료되었습니다(코드: 0개).

이 창을 닫으려면 아무 키나 누르세요...

< 설명 >

2부터 16까지 출력하는 코드이므로 반복문 while을 사용하여 간략하게 작성한다. 임의의 변수 a를 2로 초기화하고 while문의 조건을 a가 16이하일때로 걸어두고 a를 먼저 출력하고 이후 a에 2를 더해준다. 그렇게 되면 a가 2씩 증가하며 16까지 출력을 한다.

8. How many asterisks does the following code fragment print?

```
int a = 0;
while (a < 100) {
        int b = 0;
        while (b < 55) {
                std::cout << "*";
```

b++;

}

std::cout << '₩n'; }

: 한줄에 55개씩 무한대로 출력된다. 변수 a가 0으로 고정된 상태이므로 첫번째 while문은 무조건 true가 되어 계속 실행될 것이다. 첫번째 while문 안에서 변수 b를 0으로 초기화한다. 이후 두번째 while문이 실행된다. 이 안에는 * 한 개를 출력하는 코드와 이후부터 b를 1 증가시키는 코드를 포함하고 있다. 따라서, b가 55가 될 때까지 *을 출력하고 while문을 벗어나 마지막에 있는 코드 '₩n'를 출력하기 때문에 다음줄로 넘어가게 된다. 하지만 a의 값에는 변화가 없으므로 첫번째 while문은 계속 실행되어 한줄에 55개씩 출력되면서 무한줄 출력되는 것이다.

12. What is printed by the following code fragment?

int a = 0;

while (a < 100)

std::cout << a++;

std::cout << '₩n';

: < 실행결과 >

0123456789101112131415161718192021222324252627282930313233343536373839404142434445464748495051525354555657585960616263646566676869707172737475767778798081828384858687888990919293949596979899

C:₩Users₩Administrator₩source₩repos₩Ex002₩Debug₩Ex002.exe(프로세스 39256개)이(가

) 종료되었습니다(코드: 0개).

이 창을 닫으려면 아무 키나 누르세요...

< 설명 >

임의의 변수 a를 0으로 초기화하고 a가 100 미만일때 while문이 실행된다. While문 안에서는 a++를 출력한다. 따라서, 후위 증가이므로 0부터 출력되어 99까지 출력된 뒤, while문을 벗어나 줄바꿈한 뒤 컴파일이 끝난다.

14. Rewrite the following code fragment using a break statement and eliminating the done variable. Your code should behave identically to this code fragment.

```
bool done = false;

int n = 0, m = 100;

while (!done && n != m) {

        std::cin >> n;

        if (n < 0)

                done = true;

        std::cout << "n = " << n << '₩n';

}
```

: < 코드 >

```
#include <iostream>
int main() {
        int n = 0, m = 100;
        while (n != m) {
                std::cin >> n;
                std::cout << "n = " << n << '₩n';
                if (n < 0)
                        break;
        }
}
```

< 실행결과 예시 >

0

n = 0

-1

n = -1

C:₩Users₩Administrator₩source₩repos₩Ex002₩Debug₩Ex002.exe(프로세스 42288개)이(가

) 종료되었습니다(코드: 0개).

이 창을 닫으려면 아무 키나 누르세요...

< 설명 >

변수 done을 Boolean형 false로 초기화해두고 while문에서 done이 false이고 n이 m이 아닐 때 실행하도록 되어있다. While문 내에서 정수형 변수 n에 대한 입력을 받고 이 n이 음수일 경우에는 done의 값을 변경하여 n을 출력한 뒤 while문에서 빠져 나오도록 하였다. while문에서 빠져나오는 방법은 break를 사용하여 가능하다. N을 출력한 뒤 빠져나와야하므로 뒷코드와 순서를 바꾸고 if 문 안에서는 n이 음수일 경우 break하도록 한다. 따라서 done 변수는 필요가 없어지므로 제거해도 같은 결과가 나온다.

15. Rewrite the following code fragment so it eliminates the continue statement. Your new code's logic should be simpler than the logic of this fragment.

```
int x = 100, y;
while (x > 0) {
        std::cin >> y;
        if (y == 25) {
                x--;
                continue;
        }
        std::cin >> x;
        std::cout << "x = " << x << '₩n';
}
```

: < 코드 >

```
int main()
{
        int x = 100, y;
        while (--x > 0) {
                std::cin >> y;
                if (y != 25) {
                        std::cin >> x;
                        std::cout << "x = " << x << '₩n';
                        x++;
                }
        }
}
```

< 실행결과 예시 >

25

5

5

x = 5

0

0

x = 0

C:₩Users₩Administrator₩source₩repos₩Ex002₩Debug₩Ex002.exe(프로세스 22292개)이(가

) 종료되었습니다(코드: 0개).

이 창을 닫으려면 아무 키나 누르세요...

< 설명 >

원래 코드에서는 100으로 초기화된 변수 x와 y를 선언하고 while문 안에서 y를 먼저 입력받는다. 만약 이 y가 25라면 x—를 실행하고 continue하고, 그렇지 않으면 x를 입력받고 출력한다. 만약 이 x가 0보다 크지 않으면 실행 종료된다. 즉, 입력받은 x가 양수더라도 y를 입력받을 때 계속 25를 입력받는다면 x—가 실행되어 결국 x가 0이 되어 프로그램이 종료될 것이다. 이를 고려하여 코드를 더욱 간단하게 만들면 된다. while문의 조건을 수정하여 매번 진입시에 x가 -1 되도록 하고 이 때의 x가 0보다 큰 경우 while문을 실행한다. 이후 y를 입력받고 y가 25인 경우에는 문제 코드와 같이 x가 -1되며 계속 y만 입력받는다. 반대의 경우도 문제의 코드와 동일하게 실행된다. 따라서, 코드를 continue statement 사용하지 않고 더욱 간략하게 만들었다.

19. Write a C++ program that allows the user to enter exactly twenty double-precision floating-point values. The program then prints the sum, average (arithmetic mean), maximum, and minimum of the values entered.

: < 코드 >

```
int main()
{
        double x, sum, max, min;
        int i;
        i = 0;
        sum = 0;
        while (i < 20)
```

```
	{
		std::cin >> x;
		sum += x;

		if (i == 0)
		{
			max = x;
			min = x;
		}

		else if (x > max)
			max = x;

		else if (x < min)
			min = x;

		i++;
	}
	std::cout << "sum : " << sum << ", avg : " << sum / 20.0 << ", max : " << max << ",
min : " << min;
}
```

< 실행결과 예시 >

0.01

20.222

1

1

1

1

1

1

1

1

1

1

1

1

1

1

1

1

1

1

1

1

sum : 38.232, avg : 1.9116, max : 20.222, min : 0.01

C:₩Users₩Administrator₩source₩repos₩Ex002₩Debug₩Ex002.exe(프로세스 26188개)이(가

) 종료되었습니다(코드: 0개).

이 창을 닫으려면 아무 키나 누르세요...

< 설명 >

double형으로 x, sum, max, min 변수를 선언한다. Int형으로 i를 선언하여 카운트를 할 예정이다. I가 1씩 증가하며 20번 while문을 반복할 동안 변수 x를 계속 입력받으며 작업을 수행한다. sum변수에 입력받는 값을 계속 더하고, 첫번째 시행 때 max와 min에 입력값을 넣는다. 이후 시행 때는 x와 max, min 값을 각각 비교하여 max, min 변수에 최대, 최소값이 들어갈 수 있도록 한다. 따라서, 마지막에 sum, max, min 변수를 이용하여 합계, 평균, 최대값, 최소값을 모두 출력할 수 있다.

22. Redesign Listing 6.21 (startree.cpp) so that it draws a sideways tree pointing left; for example, if the user enters 7, the program would print * ** * **** ***** ****** ******* ****** ***** **** *** ** ***

: < 코드 >

```
#include <iostream>
int main() {
	int height; // Height of tree
	std::cout << "Enter height of tree: ";
	std::cin >> height; // Get height from user
	int row = 0; // First row, from the top, to draw
	while (row < height*2-1) { // Draw one row for every unit of height
	// Print leading spaces
		int count = 0;
		if (row < height - 1)
		{
			while (count < height - row - 1) {
				std::cout << " ";
```

```
                        count++;
                }
                count = 0;
                while (count < row + 1) {
                        std::cout << "*";
                        count++;
                }
        }

        if (row == height - 1)
        {
                count = 0;
                while (count < height)
                {
                        std::cout << "*";
                        count++;
                }
        }

        if (row >= height)
        {
                while (count <= row - height) {
                        std::cout << " ";
                        count++;
                }
                count = 0;
                while (count < height * 2 - row - 1) {
                        std::cout << "*";
                        count++;
                }
        }
        // Move cursor down to next line
        std::cout << '₩n';
        // Change to the next row
        row++;
    }
}
```

< 실행결과 >

Enter height of tree: 7

```
       *
      **
     ***
    ****
   *****
  ******
```

****

 ******

  *****

   ****

    ***

     **

      *

C:₩Users₩Administrator₩source₩repos₩Ex002₩Debug₩Ex002.exe(프로세스 59420개)이(가

) 종료되었습니다(코드: 0개).

이 창을 닫으려면 아무 키나 누르세요...

< 설명 >

Listing 6.21에 있는 것을 수정하여 코드를 작성하라고 하여 본 틀을 가져와 일부만 수정하였다. 원본에서는 아래로 내려갈수록 단순히 *의 개수가 많아지는 것이었으나 이 문제는 조금 더 심오했다. *의 개수가 증가하는 부분과 가운데, *의 개수가 감소하는 부분으로 나누어 if문을 작성하고 안에서 해당 *의 개수가 나오도록 while문을 작성하였다. 공백은 원본 코드와 거의 동일한 원리로 구현하였다.

B. Exercises (Write the questions down on your answer sheet)

(pp. 173-177), Exercises 2, 3, 5
(write output analysis for all exercises)

2. Consider the following code fragment.

```
char ch;
std::cin >> ch;
switch (ch) {
        case 'a':
                std::cout << "*₩n";
```

```
            break;
        case 'A':
        std::cout << "**₩n";
        break;
        case 'B':
        case 'b':
        std::cout << "***₩n";
        case 'C':
        case 'c':
        std::cout << "****₩n";
        break;
        default:
        std::cout << "*****₩n";
        }
```

(a) What is printed when the user enters a?

: < 실행결과 >

a

*

C:₩Users₩Administrator₩source₩repos₩Ex002₩Debug₩Ex002.exe(프로세스 66696개)이(가

) 종료되었습니다(코드: 0개).

이 창을 닫으려면 아무 키나 누르세요...

- switch문에서 case 'a'에 해당하기 때문에 그 안에 있는 std::cout<< "*₩n";이 실행된다.

(b) What is printed when the user enters A?

< 실행결과 >

A

**

C:₩Users₩Administrator₩source₩repos₩Ex002₩Debug₩Ex002.exe(프로세스 62416개)이(가

) 종료되었습니다(코드: 0개).

이 창을 닫으려면 아무 키나 누르세요...

- switch문에서 case 'A'에 해당하기 때문에 그 안에 있는 std::cout<< "**₩n";이 실행된다.

(c) What is printed when the user enters b?

: < 실행결과 >

b

***

****

C:₩Users₩Administrator₩source₩repos₩Ex002₩Debug₩Ex002.exe(프로세스 59824개)이(가

) 종료되었습니다(코드: 0개).

이 창을 닫으려면 아무 키나 누르세요...

- switch문에서 case 'b'에 해당하기 때문에 그 안에 있는 std::cout<< "***₩n";이 실행된다. 그러나 break 되지 않았기 때문에 break 되기까지의 그 이후 코드가 전부 실행되어 std::cout<< "****₩n"; 도 실행된다.

(d) What is printed when the user enters B?

: < 실행결과 >

B

***

****

C:₩Users₩Administrator₩source₩repos₩Ex002₩Debug₩Ex002.exe(프로세스 68356개)이(가

) 종료되었습니다(코드: 0개).

이 창을 닫으려면 아무 키나 누르세요...

- (c)와 동일하다.

(e) What is printed when the user enters C?

: < 실행결과 >

C

****

C:₩Users₩Administrator₩source₩repos₩Ex002₩Debug₩Ex002.exe(프로세스 67924개)이(가

) 종료되었습니다(코드: 0개).

이 창을 닫으려면 아무 키나 누르세요...

- switch문에서 case 'C'에 해당하기 때문에 그 안에 있는 std::cout<< "****₩n";이 실행된다.

(f) What is printed when the user enters c?

: < 실행결과 >

c

****

C:₩Users₩Administrator₩source₩repos₩Ex002₩Debug₩Ex002.exe(프로세스 67860개)이(가

) 종료되었습니다(코드: 0개).

이 창을 닫으려면 아무 키나 누르세요...

- switch문에서 case 'c'에 해당하기 때문에 그 안에 있는 std::cout<< "****₩n";이 실행된다.

(g) What is printed when the user enters t?

: t

*****

C:₩Users₩Administrator₩source₩repos₩Ex002₩Debug₩Ex002.exe(프로세스 65864개)이(가

) 종료되었습니다(코드: 0개).

이 창을 닫으려면 아무 키나 누르세요...

- switch문에서 default에 해당하기 때문에 그 안에 있는 std::cout<< "*****₩n";이 실행된다.

3. What is printed by the following code fragment?

```
int x = 0;
do {
std::cout << x << " ";
x++;
} while (x < 10);
std::cout << '₩n';
```

: < 실행결과 >

0 1 2 3 4 5 6 7 8 9

C:₩Users₩Administrator₩source₩repos₩Ex002₩Debug₩Ex002.exe(프로세스 44812개)이(가

) 종료되었습니다(코드: 0개).

이 창을 닫으려면 아무 키나 누르세요...

< 설명 >

변수 x가 정수형으로 선언, 0으로 초기화. Do while문을 통해 반복할 예정. 루프 조건이 초기에 참이므로 (x<10) 거짓이 될 때 까지 루프를 돈다. 따라서 x가 9가 될때까지 1씩 증가하며 출력되는 것이다.

5. What is printed by the following code fragment?

```
for (int x = 0; x < 10; x++)
        std::cout << "*";
std::cout << '₩n';
```

: < 실행결과 >

****

C:₩Users₩Administrator₩source₩repos₩Ex002₩Debug₩Ex002.exe(프로세스 68832개)이(가

) 종료되었습니다(코드: 0개).

이 창을 닫으려면 아무 키나 누르세요...

< 설명 >

For 반복문에서는 초기상태, 조건, 변화식을 받아 실행하는 것이다. 초기식이 x=0이고 조건이 x<10이므로 변화식인 x++에 따라 x를 0부터 시작하여 1씩 증가하다가 조건이 참일동안 for문 안의 내용을 실행하는 것이다. 따라서 x가 0~9가 되는 동안 *이 10개 출력되는 것이다.

C. Additional exercises (Write the questions down on your answer sheet)

C-1. Euler's number, e, is used as the base of natural logarithms. It can be approximated using the following formula.

$$e = 1 + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \frac{1}{4!} + \cdots$$

Write a program that approximate e using a loop that terminates when the difference between two successive values of e is less than 0.00001.

: < 코드 >

```
int main() {
	double e = 1, n1 = 1, n2 = 1, e1 = 1, e2 = 2;
	for (int i = 1; e > 0.00001; i++)
	{
		n1 *= i;
		n2 *= i + 1;
		e1 += 1 / n1;
		e2 += 1 / n2;
		e = e2 - e1;
	}
	std::cout << "분모의 n이 " << n2 << "일 때, e가 0.00001보다 작은 " << e << "이다.";
}
```

< 실행결과 >

분모의 n이 362880일 때, e가 0.00001보다 작은 2.75573e-06이다.

C:₩Users₩Administrator₩source₩repos₩Ex002₩Debug₩Ex002.exe(프로세스 24800개)이(가

) 종료되었습니다(코드: 0개).

이 창을 닫으려면 아무 키나 누르세요...

< 설명 >

E가 0.00001보다 작아지면 루프를 바로 벗어나오게 구성해둔다. 오일러 공식을 구현하기 위한 변수들을 선언하고 초기화한다. N을 계속 늘려가면서 공식을 돌리다가 두 근접한 값의 차가 0.00001보다 작아졌을 경우 루프를 벗어나와 그때의 n값과 e값을 출력한다.

C-2. Write a program that reads a positive integer less than 1,000,000 from the keyboard and then prints it out in reverse. For example, if the user enters 2576, it prints 6752.

: < 코드 >

```
#include <iostream>
int main() {
	int num, temp;
	std::cin >> num;
	if (num < 10)
		std::cout << num;
	else
	{
		while (!(num < 10))
		{
			temp = num % 10;
			num = num / 10;
			std::cout << temp;
		}
		std::cout << num;
	}
}
```

< 실행결과 >

2576

6752

C:₩Users₩Administrator₩source₩repos₩Ex002₩Debug₩Ex002.exe(프로세스 78824개)이(가

) 종료되었습니다(코드: 0개).

이 창을 닫으려면 아무 키나 누르세요...

< 설명 >

while문을 통해 자릿수를 변환한다고 생각하고 코드를 구성함. 변수를 입력 받고 이 변수가 한자리수가 될 때까지 10으로 나눈 나머지를 먼저 출력하면서 자릿수를 변환한다음, 마지막에 한자리수가 되었을 때 그대로 이어서 출력을 하면 자릿수를 거꾸로 하는 것이 가능하다.

* Write a `for` loop that will produce each of following sequences:

C-3. 6, 8, 10, 12, ..., 60

: < 코드 >

```
#include <iostream>
int main() {
        for (int i = 6; i <= 60; i += 2)
                std::cout << i << " ";
}
```

< 실행결과 >

6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50 52 54 56 58 60

C:₩Users₩Administrator₩source₩repos₩Ex002₩Debug₩Ex002.exe(프로세스 75548개)이(가

) 종료되었습니다(코드: 0개).

이 창을 닫으려면 아무 키나 누르세요...

< 설명 >

초기조건을 정수형 I = 6으로 두고 i가 60이 될 때까지 2씩 증가시키면서 계속 출력한다.

C-4. 7, 9, 11, 13, ..., 67

: < 코드 >

```
#include <iostream>
int main() {
        for (int i = 7; i <= 67; i += 2)
                std::cout << i << " ";
}
```

< 실행결과 >

7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57 59 61 63 65 67

C:₩Users₩Administrator₩source₩repos₩Ex002₩Debug₩Ex002.exe(프로세스 72600개)이(가

) 종료되었습니다(코드: 0개).

이 창을 닫으려면 아무 키나 누르세요...

< 설명 >

위와 마찬가지로 초기조건을 정수형 I = 7로 두고 I가 67이 될 때까지 2씩 증가시키면서 계속 출

력한다.

C-5. The sum of the numbers between 1 and 15 inclusive.

: < 코드 >

```
#include <iostream>
int main() {
	int sum = 0;
	for (int i = 1; i <= 15; i++)
		sum += i;
	std::cout << sum;
}
```

< 실행결과 >

120

C:₩Users₩Administrator₩source₩repos₩Ex002₩Debug₩Ex002.exe(프로세스 69564개)이(가

) 종료되었습니다(코드: 0개).

이 창을 닫으려면 아무 키나 누르세요...

< 설명 >

1과 15를 포함한 사이의 숫자들의 합이므로 i를 1부터 15까지 1씩 증가시키면서 반복하며 sum 변수에 더해주면 된다.

C-6. The first 50 numbers in the series, 1, 4, 7, 10, ...

< 코드 >

```
#include <iostream>
int main() {
	int cnt = 0;
	for (int i = 1; cnt < 50; cnt++)
	{
		std::cout << i << " ";
		i += 3;
	}
}
```

< 실행결과 >

1 4 7 10 13 16 19 22 25 28 31 34 37 40 43 46 49 52 55 58 61 64 67 70 73 76 79 82

85 88 91 94 97 100 103 106 109 112 115 118 121 124 127 130 133 136 139 142 145

148

C:₩Users₩Administrator₩source₩repos₩Ex002₩Debug₩Ex002.exe(프로세스 82556개)이(가

) 종료되었습니다(코드: 0개).

이 창을 닫으려면 아무 키나 누르세요...

< 설명 >

변수를 하나만 써서도 구현이 가능하지만, 그렇다면 이전 문제와 다를게 없어보여 이번에는 변수를 두개 사용하여 for문을 구사해보았다. 1부터 3씩 더해지는 50개의 숫자 나열이므로 cnt변수를 이용해 50번을 카운트 하며 i에 3씩 더하는 것을 반복하며 차례대로 출력한다.