

# 과 제

## Programming Assignment 1



과목명	C프로그래밍
분반	0451분반
담당교수님	박소영 교수님
학과	컴퓨터공학부
학번	202011353
이름	이호은
제출일	2020년 4월 29일

# Programming Assignment 1

## Question 1. 사각 테이블 배치 문제

### 1) 문제 정의

방의 가로, 세로 크기, 테이블의 가로, 세로 크기, 테이블 간 간격, 테이블 당 하객 수를 입력 받고 방과의 테이블 간격, 테이블 사이의 간격을 만족했을 때의 최대 수용 인원을 계산하여 출력한다.

입력 조건	출력 조건
<div>- 방의 가로 및 세로 크기는 미터 단위로 양의 정수이며 1000보다 는 작다고 가정한다.</div> <div>- 테이블의 가로 및 세로 크기는 미터 단위로 양의 정수이며 1000 보다는 작다고 가정한다.</div> <div>- 테이블 간 간격은 미터 단위로 양의 정수이며 10보다 작다고 가 정한다.</div> <div>- 테이블당 하객 수는 양의 정수로 3명이상 20명 이하라고 가정한 다.</div> <div>- 각 입력 값의 범위는 프로그램에서 검사할 필요는 없으며, 사용자 가 값을 입력할 때, 주어진 조건에 맞는 값을 입력한다고 가정한다. 단, 변수 선언을 위한 자료형은 입력 값의 범위를 고려하여, 최소의 메모리를 사용할 수 있도록 선언한다.</div>	<div>프로그램은 주어진 조건에 따른 최대 하객 수를 출력한다. 결과는 반드시 다음의 출력 형태를 따르며, X로 표시된 부분은 프로그램 실행 에 의해 계산된 값이다.</div> <div>This arrangement seats X people.</div>

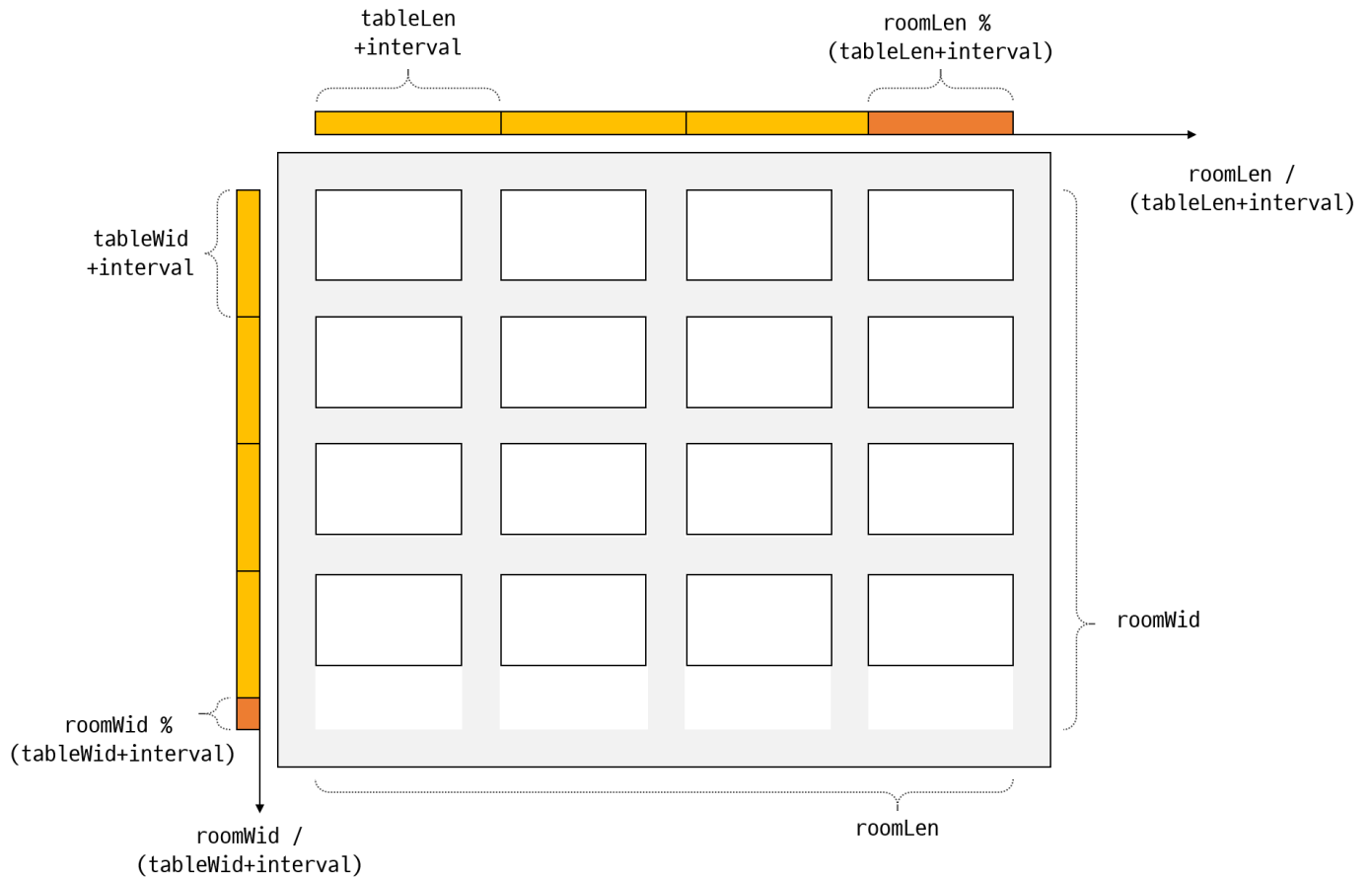
### 2) 주요 변수 설명

변수명	자료형	설명
roomLen	int	방의 가로 길이(length)를 저장한다. (m)
roomWid	int	방의 세로 길이(width)를 저장한다. (m)
tableLen	int	테이블 한 개의 가로 길이(length)를 저장한다. (m)
tableWid	int	테이블 한 개의 세로 길이(width)를 저장한다. (m)
interval	unsigned char	테이블과 테이블 사이, 방 끝과 테이블 사이의 간격을 저장한다. (m)
person	int	테이블 당 사람 수, 나중에는 이 변수값과 테이블 수를 곱하여 전체 사람 수를 저장한다.
temp	int	변수 interval의 값을 입력 받을 때 사용.*

- unsigned char의 범위는 0~255까지의 수를 저장할 수 있으며 정수를 저장하는 변수 중에선 가장 크기가 작다. 즉, 범위가 1~10인 interval과 3~20인 person의 값을 저장하기에 용이하다. 하지만 person 변수는 나중에 최종적인 하객 수를 저장하게 되므로 255를 넘어갈 수 있으므로 정수형으로 선언해주었다.

\* scanf("%d", &interval); 과 같이 unsigned char 변수에 scanf 함수로 10진수 값을 입력 받아 저장하자 프로그램 실행 도중 에러가 발생하였습니다. 이를 해결하기 위해 정수형 변수에 값을 입력 받은 후 이 값을 다시 각 변수에 대입해주는 방식으로 해결해주었다.

### 3) 아이디어 및 알고리즘



- 총 테이블의 수는 가로로 배치되는 최대 테이블의 수와 세로로 배치되는 최대 테이블 수를 곱한 값이다. 총 하객 수는 총 테이블 수에 테이블 당 사람 수를 곱한 값이다.
- 먼저 방의 사면의 벽에는 테이블과 간격이 있어야 한다. 실질적으로 테이블이 놓일 수 있는 방의 크기는 가로  $roomLen - 2 * interval$ , 세로는  $roomWid - 2 * interval$ 이다.
- 실질적 방의 테이블이 가로, 세로로 각각 몇 개가 들어가는지 확인하기 위해서는 테이블의 가로, 세로 길이와 간격을 더한 값을 하나의 테이블로 생각하면 간격이 있을 때의 테이블의 최대 개수를 구할 수 있다.
- $roomWid / (tableWid + interval)$ 과  $roomLen / (tableLen + interval)$ 로 가로, 세로로 들어가는 테이블의 수를 구할 수 있다.(여기서  $roomWid$ 와  $roomWid$ 는 실질적 방의 가로, 세로 길이이다. 즉  $roomLen = roomLen - 2interval$ ,  $roomWid = roomWid - 2interval$ 이다.)
- 여기서 문제가 각 마지막 줄에 배치되는 테이블의 경우 간격을 더하게 되면 끝쪽에는 방으로부터의 간격과 테이블로부터의 간격, 즉 간격이 2개가 생기게 된다. 즉 남은 공간 ( $방의\ 길이 / (테이블\ 길이 + 간격)$ )의 나머지 즉, ( $방의\ 길이 \% (테이블의\ 길이 + 간격)$ ) 내에 테이블이 들어갈 수 있는지를 확인해야 한다.
- 이때, 테이블의 길이보다 나머지가 크다면 몫이 +1이 될 것이므로 나머지의 최댓값은 테이블의 길이와 같다. 즉, 나머지 공간에 테이블이 한 줄 더 들어갈 수 있는 경우는 ( $나머지 = (방의\ 길이 \% (테이블의\ 길이 + 간격)) == 테이블의\ 길이$ )가 될 것이다. 이때, 만약 나머지 공간에 테이블이 들어갈 수 있다면 조건식이 참이 되어 +1이 될 것이고 테이블이 들어갈 수 없다면 조건식이 거짓이 되어 +0이 될 것이다.

## - 알고리즘

- ㉠ 데이터를 입력 받는다.
- ㉡ 실제로 테이블이 배치될 수 있는 방의 가로, 세로 길이를 각각 구한다. = (방 길이 - 2 \* 간격)
- ㉢ 실질적 방 가로 길이 / (테이블 가로 길이 + 간격)
- ㉣ 만약 ㉢에서 테이블이 배치되고 남은 길이에 테이블이 들어갈 수 있다면 +1
- ㉤ 이를 person 변수의 값과 곱하여 person 변수에 대입한다. (사람수 x 가로 개수)
- ㉥ 실질적 방 세로 길이 / (테이블 세로 길이 + 간격)
- ㉦ 만약 ㉥에서 테이블이 배치되고 남은 길이에 테이블이 들어갈 수 있다면 +1
- ㉧ 이를 person 변수의 값과 곱하여 person 변수에 대입한다. (사람수 x 가로 개수 x 세로 개수)
- ㉨ 결과를 출력한다.

## 4) 소스코드

```
#include <stdio.h>

/* Programming Assignment I *
 * 0451분반 202011353 이호은 */

int main(void) {

    /* 변수 선언 */
    int roomLen, roomWid;      // 방의 가로, 세로 길이(m)
    int tableLen, tableWid;    // 테이블의 가로, 세로 길이(m)
    int temp, person;          // temp 변수, 테이블 당 사람 수
    unsigned char interval;     // 테이블 사이 간격(m)

    /* 데이터 입력 */
    printf("What are the length and width of the room (in meters)?\n");
    scanf("%d %d", &roomLen, &roomWid);

    printf("What are the length and width of each table (in meters)?\n");
    scanf("%d %d", &tableLen, &tableWid);

    printf("How much space is required between tables (in meters)?\n");
    scanf("%d", &temp);
    interval = temp;

    printf("How many people does each table seat?\n");
    scanf("%d", &person);

    /* 테이블이 배치될 수 있는 실질적 가로, 세로 길이 */
    roomLen -= 2 * interval;
    roomWid -= 2 * interval;

    /* 방의 길이 / (테이블 길이 + 간격) 만약 남은 길이에 테이블이 1개 들어간다면 +1, 들어가지 않는다면 +0 */
    person *= (roomLen / (tableLen + interval)) + ((roomLen % (tableLen + interval)) == tableLen);
    person *= (roomWid / (tableWid + interval)) + ((roomWid % (tableWid + interval)) == tableWid);

    /* 결과 출력 */
    printf("This arrangement seats %d people.\n", person);

    return 0;
}
```

## 5) 수행 결과

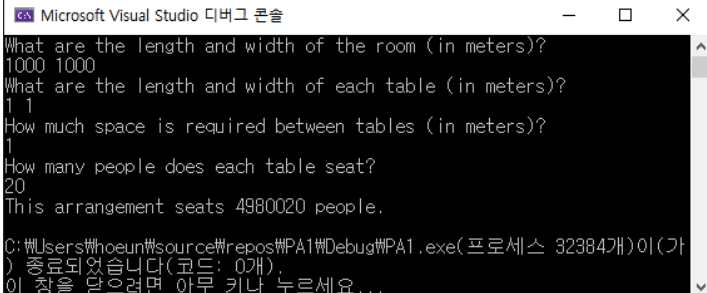
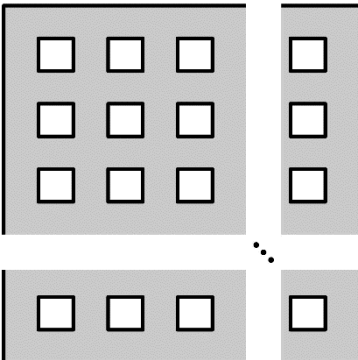
- 주어진 Sample Data를 이용한 수행

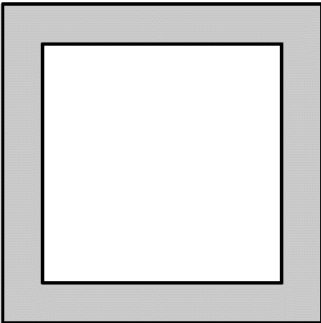
```

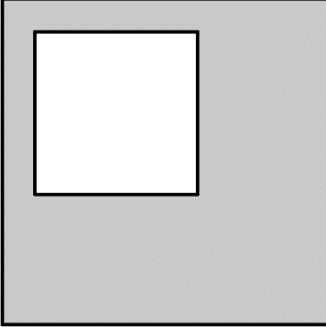
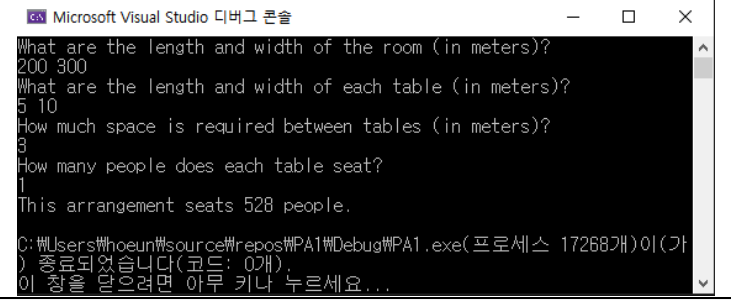
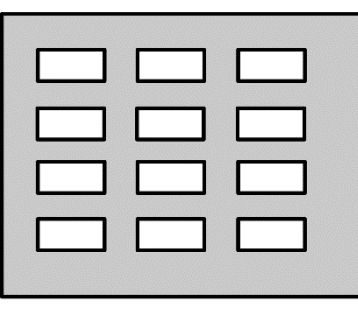
Microsoft Visual Studio 디버그 콘솔
What are the length and width of the room (in meters)?
50 30
What are the length and width of each table (in meters)?
8 4
How much space is required between tables (in meters)?
3
How many people does each table seat?
10
This arrangement seats 120 people.

C:\Users\hoeun\source\repos\PA1\Debug\PA1.exe(프로세스 11492개)이(가)
) 종료되었습니다(코드: 0개).
이 창을 닫으려면 아무 키나 누르세요...
  
```

- 데이터 셋을 통한 프로그램 검증

데이터 설정	
1	int roomLen = 1000, roomWid = 1000 int tableLen = 1, tableWid = 1 int interval = 1, person = 20
	프로그램 출력 결과
	
	4,980,020명
검증	
(최대의 경우) 	
1000m x 1000m의 방 안에 1m x 1m의 테이블이 1m 간격으로 배치되면 이와 같은 모습이다. 가로로 테이블은 간격, 테이블, 간격, 테이블, ..., 테이블, 간격 순으로 놓이게 된다. 1000m일 때 1m의 간격을 두고 1m 길이의 테이블은 499개가 들어갈 수 있다. (998 / 2) 가로 499개 x 세로 499개 = 249,001개이다. 249,001개의 테이블에 20명씩 앉는다면 249,001개 x 20명 = 4,980,020명이다.	
데이터 설정	
2	int roomLen = 100, roomWid = 100 int tableLen = 80, tableWid = 80 int interval = 10, person = 1
	프로그램 출력 결과

	<div><div>Microsoft Visual Studio 디버그 콘솔</div><div>What are the length and width of the room (in meters)? 100 100 What are the length and width of each table (in meters)? 80 80 How much space is required between tables (in meters)? 10 How many people does each table seat? 1 This arrangement seats 1 people. C:\Users\hnoeun\source\repos\PA1\Debug\PA1.exe(프로세스 8900개)이(가) 종료되었습니다(코드: 0개). 이 창을 닫으려면 아무 키나 누르세요...</div></div>	1명
검증		
(정확히 일치하는 경우)		
	<p>100m x 100m의 방 안에 80m x 80m 테이블이 10m 간격으로 배치되면 이와 같은 모습이다.</p> <p>실제로 테이블이 배치될 수 있는 공간의 크기는 가로, 세로 길이가 <math>100\text{m} - 2 * 10\text{m} = 80\text{m}</math>이다.</p> <p>80m x 80m의 공간 안에 80m x 80m 크기의 1개 테이블이 들어갈 수 있다. (정확히 일치)</p> <p>1개의 테이블에 1명이 앉는다면 1명이다.</p>	
데이터 설정		
<pre>int roomLen = 10, roomWid = 10 int tableLen = 100, tableWid = 100 int interval = 10, person = 10</pre>		
프로그램 출력 결과		
3	<div><div>Microsoft Visual Studio 디버그 콘솔</div><div>What are the length and width of the room (in meters)? 10 10 What are the length and width of each table (in meters)? 100 100 How much space is required between tables (in meters)? 10 How many people does each table seat? 10 This arrangement seats 0 people. C:\Users\hnoeun\source\repos\PA1\Debug\PA1.exe(프로세스 17100개)이(가) 종료되었습니다(코드: 0개). 이 창을 닫으려면 아무 키나 누르세요...</div></div>	0명
검증		
(초과의 경우)		
<p>10m x 10m 방에 100m x 100m 크기의 테이블이 들어갈 수 없으므로 0명이다.</p>		
데이터 설정		
<pre>int roomLen = 100, roomWid = 100 int tableLen = 50, tableWid = 50 int interval = 10, person = 1</pre>		
프로그램 출력 결과		
4	<div><div>Microsoft Visual Studio 디버그 콘솔</div><div>What are the length and width of the room (in meters)? 100 100 What are the length and width of each table (in meters)? 50 50 How much space is required between tables (in meters)? 10 How many people does each table seat? 1 This arrangement seats 1 people. C:\Users\hnoeun\source\repos\PA1\Debug\PA1.exe(프로세스 8264개)이(가) 종료되었습니다(코드: 0개). 이 창을 닫으려면 아무 키나 누르세요...</div></div>	1명
검증		

		<p>100m x 100m의 방 안에 50m x 50m 테이블이 10m 간격으로 배치되면 이와 같은 모습이다.</p> <p>실제로 테이블이 배치될 수 있는 공간의 크기는 가로, 세로 길이가 <math>100\text{m} - 2 * 10\text{m} = 80\text{m}</math>이다.</p> <p>80m x 80m의 공간 안에 50m x 50m 크기의 1개 테이블이 들어갈 수 있다. (남은 공간의 가로, 세로 길이는 30m 이므로 테이블이 들어갈 수 없다.)</p> <p>1개의 테이블에 1명이 앉는다면 1명이다.</p>
5	데이터 설정	
	<pre>int roomLen = 200, roomWid = 300 int tableLen = 5, tableWid = 10 int interval = 3, person = 1</pre>	
	프로그램 출력 결과	
		528명
검증		
		<p>가로의 경우 <math>194\text{m}(200\text{m} - 2*3\text{m})</math> 안에 5m 길이의 테이블이 3m의 간격을 두고 24개가 들어갈 수 있다. <math>(194 / (5 + 3))</math> 나머지는 2m이므로 5m의 테이블이 들어갈 수 없다.</p> <p>세로의 경우 <math>294\text{m}(300\text{m} - 2*3\text{m})</math> 안에 10m 길이의 테이블이 3m의 간격을 두고 22개가 들어갈 수 있다. <math>(294 / (10 + 3))</math> 나머지는 8m이므로 10m의 테이블이 들어갈 수 없다.</p> <p>가로 24개 x 세로 22개 = 528개이다.</p>

## 6) 토의 사항

- 그동안 프로그래밍을 하면서 연산식 내에 조건식(부등식, 등식 등)을 이용해서 연산을 한 적은 없었는데 이번 프로그래밍을 하며 연산식 내에 조건식을 도입한다면 더욱 연산이 간단해진다는 것을 배웠습니다.
- 사실 이번 코딩을 하며 거의 1주일 가까이 시간이 걸렸습니다. 그동안 프로그래밍을 하며 수학적인 연산에 관련된 프로그래밍을 많이 접해보았지만 거의 수학 자체에 가까운 프로그램이다보니 알고리즘을 여러 번 수정하며 프로그램을 짰습니다. 많은 시도 끝에 돌고 돌아 코드가 처음과 거의 유사한 형태가 되어서 약간 놀라기도 했습니다. 프로그래밍을 할 때, 문법 요소를 잘 아는 것도 중요하지만 얼마나 수학적 알고리즘을 효율적으로 빠르게 짤 수 있느냐가 중요하다는 것을 배웠습니다.

## Question 2. 원탁 테이블 배치 문제

### 1) 문제 정의

방의 가로, 세로 크기, 테이블의 반지름 길이, 테이블 간 간격, 테이블 당 하객 수를 입력 받고 방과의 테이블 간격, 테이블 사이의 간격을 만족했을 때의 최대 수용 인원을 계산하여 출력한다.

입력 조건	출력 조건
<ul style="list-style-type: none"> <li>- 방의 가로 및 세로 크기는 미터 단위로 양의 정수이며 1000보다 작다고 가정한다.</li> <li>- 테이블의 반지름의 길이는 미터 단위로 양의 정수이며, 1에서 500 사이의 값이다.</li> <li>- 테이블 간 간격은 미터 단위로 양의 정수이며 10보다 작다고 가정한다.</li> <li>- 테이블당 하객 수는 양의 정수로 3명 이상 20명 이하라고 가정한다.</li> <li>- 각 입력 값의 범위는 프로그램에서 검사할 필요는 없으며, 사용자가 값을 입력할 때, 주어진 조건에 맞는 값을 입력한다고 가정한다. 단, 변수 선언을 위한 자료형은 입력 값의 범위를 고려하여, 최소의 메모리를 사용할 수 있도록 선언한다.</li> </ul>	<p>프로그램은 주어진 조건에 따른 최대 하객 수를 출력한다. 결과는 반드시 다음의 출력 형태를 따르며, X로 표시된 부분은 프로그램 실행에 의해 계산된 값이다.</p> <p>This arrangement seats X people.</p>

### 2) 주요 변수 설명

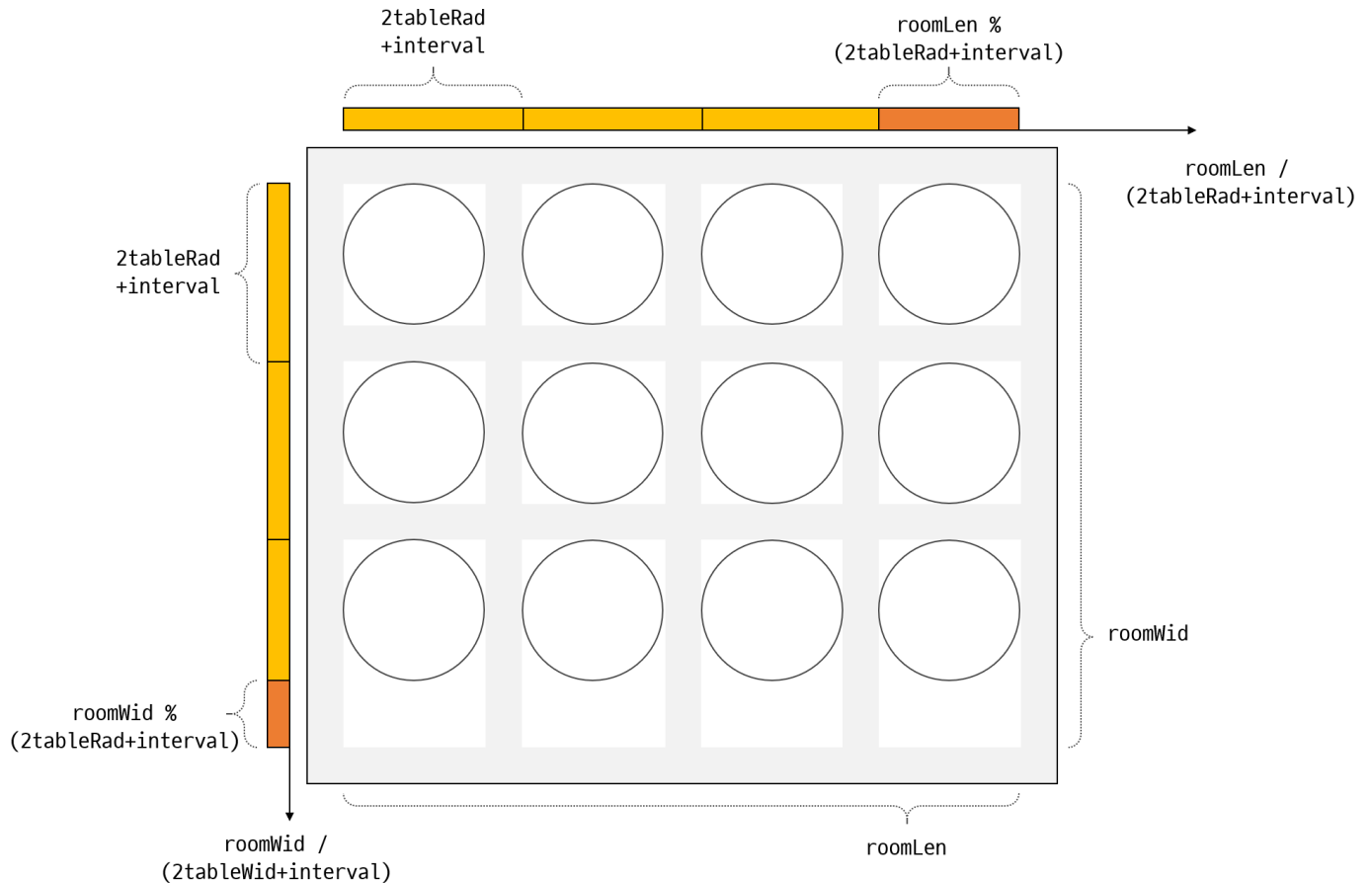
변수명	자료형	설명
<b>roomLen</b>	int	방의 가로 길이(length)를 저장한다. (m)
<b>roomWid</b>	int	방의 세로 길이(width)를 저장한다. (m)
<b>tableRad</b>	int	테이블 한 개의 반지름 길이를 저장한다. (m)
<b>interval</b>	unsigned char	테이블과 테이블 사이, 방 끝과 테이블 사이의 간격을 저장한다. (m)
<b>person</b>	int	테이블 당 사람 수, 나중에는 이 변수값과 테이블 수를 곱하여 전체 사람 수를 저장한다.
<b>temp</b>	int	변수 interval의 값을 입력 받을 때 사용.*

- unsigned char의 범위는 0~255까지의 수를 저장할 수 있으며 정수를 저장하는 변수 중에선 가장 크기가 작다. 즉, 범위가 1~10인 **interval**과 3~20인 **person**의 값을 저장하기에 용이하다. 하지만 **person** 변수는 나중에 최종적인 하객 수를 저장하게 되므로 255를 넘어갈 수 있으므로 정수형으로 선언해주었다.

\* `scanf("%d", &interval);` 과 같이 unsigned char 변수에 scanf 함수로 10진수 값을 입력 받아 저장하자 프로그램 실행 도중 에러가 발생하였습니다. 이를 해결하기 위해 정수형 변수에 값을 입력 받은 후 이 값을 다시 각 변수에 대입해주는 방식으로 해결해주었다.



### 3) 아이디어 및 알고리즘



- 총 테이블의 수는 가로로 배치되는 최대 테이블의 수와 세로로 배치되는 최대 테이블 수를 곱한 값이다. 총 하객 수는 총 테이블 수에 테이블 당 사람 수를 곱한 값이다.
- 먼저 방의 사면의 벽에는 테이블과 간격이 있어야 한다. 실질적으로 테이블이 놓일 수 있는 방의 크기는 가로  $roomLen - 2 * interval$ , 세로는  $roomWid - 2 * interval$ 이다.
- 1번 문제인 사각 테이블 배치 문제와 알고리즘은 일치한다. 다만 테이블의 가로, 세로 길이가 아닌 테이블의 지름으로 계산해야 한다.
- 실질적 방의 테이블이 가로, 세로로 각각 몇 개가 들어가는지 확인하기 위해서는 테이블의 지름 길이와 간격을 더한 값을 하나의 테이블로 생각하면 간격이 있을 때의 테이블의 최대 개수를 구할 수 있다.
- $roomWid / (tableRad + interval)$ 과  $roomLen / (tableRad + interval)$ 로 가로, 세로로 들어가는 테이블의 수를 구할 수 있다.(여기서  $roomWid$ 와  $roomWid$ 는 실질적 방의 가로, 세로 길이이다. 즉  $roomLen = roomLen - 2*interval$ ,  $roomWid = roomWid - 2*interval$ 이다.)
- 여기서 문제가 각 마지막 줄에 배치되는 테이블의 경우 간격을 더하게 되면 끝쪽에는 방으로부터의 간격과 테이블로부터의 간격, 즉 간격이 2개가 생기게 된다. 즉 남은 공간 ( $방의\ 길이 / (테이블\ 지름\ 길이 + 간격)$ )의 나머지가, 즉, ( $방의\ 길이 \% (테이블의\ 지름\ 길이 + 간격)$ ) 내에 테이블이 들어갈 수 있는지를 확인해야 한다.
- 이때, 테이블의 길이보다 나머지가 크다면 몫이 +1이 될 것이므로 나머지의 최댓값은 테이블의 길이와 같다. 즉, 나머지 공간에 테이블이 한 줄 더 들어갈 수 있는 경우는 ( $나머지 = (방의\ 길이 \% (테이블의\ 지름\ 길이 + 간격)) == 테이블의\ 지름\ 길이$ )가 될 것이다. 이때, 만약 나머지 공간에 테이블이 들어갈 수 있다면 조건식이 참이 되어 +1이 될 것이고 테이블이 들어갈 수 없다면 조건식이 거짓이 되어 +0이 될 것이다.

## - 알고리즘

- ㉑ 데이터를 입력 받는다.
- ㉒ 실제로 테이블이 배치될 수 있는 방의 가로, 세로 길이를 각각 구한다. = (방 길이 - 2 \* 간격)
- ㉓ 실질적 방 가로 길이 / (테이블 지름 길이 + 간격)
- ㉔ 만약 ㉓에서 테이블이 배치되고 남은 길이에 테이블이 들어갈 수 있다면 +1
- ㉕ 이를 person 변수의 값과 곱하여 person 변수에 대입한다. (사람수 x 가로 개수)
- ㉖ 실질적 방 세로 길이 / (테이블 지름 길이 + 간격)
- ㉗ 만약 ㉖에서 테이블이 배치되고 남은 길이에 테이블이 들어갈 수 있다면 +1
- ㉘ 이를 person 변수의 값과 곱하여 person 변수에 대입한다. (사람수 x 가로 개수 x 세로 개수)
- ㉙ 결과를 출력한다.

## 4) 소스코드

```
#include <stdio.h>

/* Programming Assignment I *
 * 0451분반 202011353 이호은 */

int main(void) {

    /* 변수 선언 */
    int roomLen, roomWid;      // 방의 가로, 세로 길이(m)
    int tableRad;              // 테이블의 반지름(m)
    int temp, person;          // temp 변수, 테이블 당 사람 수(명)
    unsigned char interval;    // 테이블 사이 간격(m)

    /* 데이터 입력 */
    printf("What are the length and width of the room (in meters)?\n");
    scanf("%d %d", &roomLen, &roomWid);

    printf("What are the radius of each table (in meters)?\n");
    scanf("%d", &tableRad);

    printf("How much space is required between tables (in meters)?\n");
    scanf("%d", &temp);
    interval = temp;

    printf("How many people does each table seat?\n");
    scanf("%d", &person);

    /* 테이블이 배치될 수 있는 실질적 가로, 세로 길이 */
    roomLen -= 2 * interval;
    roomWid -= 2 * interval;

    /* 방의 길이 / (테이블 길이 + 간격) 만약 남은 길이에 테이블이 1개 들어간다면 +1, 들어가지 않는다면 +0 */
    person *= (roomLen / (2*tableRad + interval)) + ((roomLen % (2*tableRad + interval)) == 2*tableRad);
    person *= (roomWid / (2*tableRad + interval)) + ((roomWid % (2*tableRad + interval)) == 2*tableRad);

    /* 결과 출력 */
    printf("This arrangement seats %d people.\n", person);

    return 0;
}
```

## 5) 수행 결과

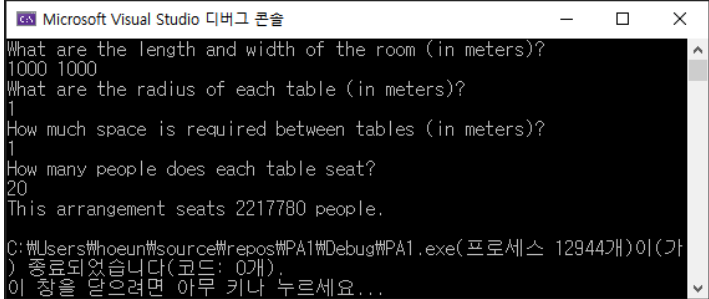
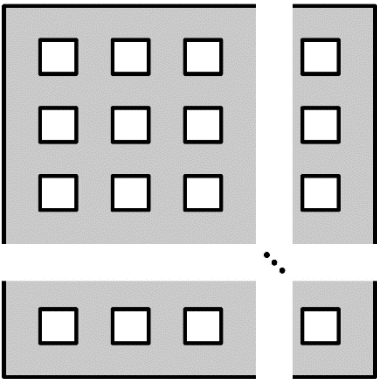
- 주어진 Sample Data를 이용한 수행

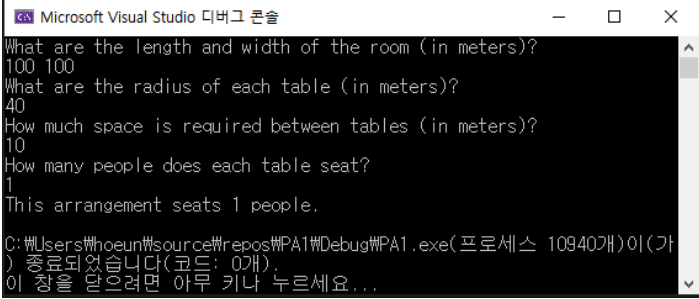
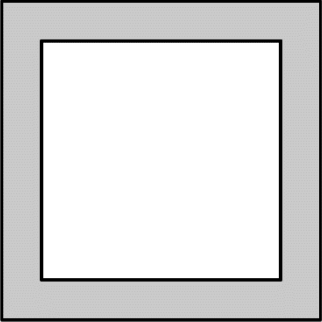
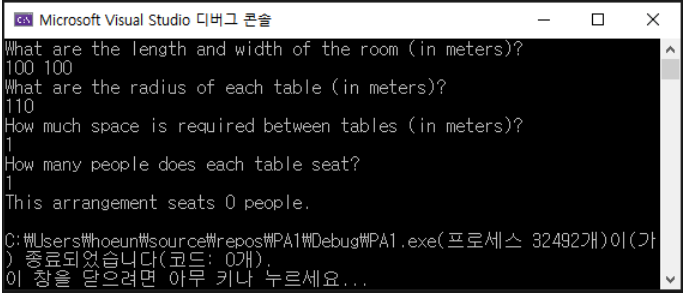
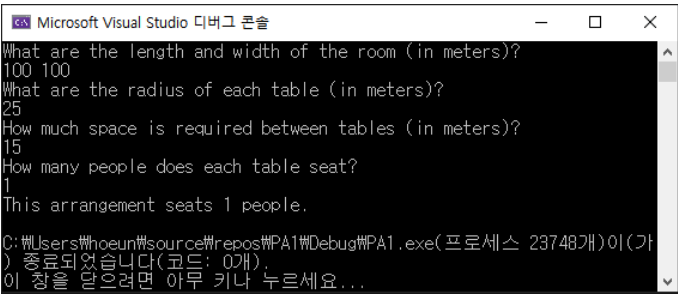
```

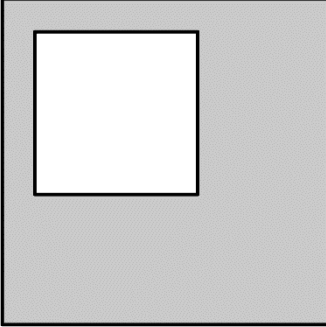
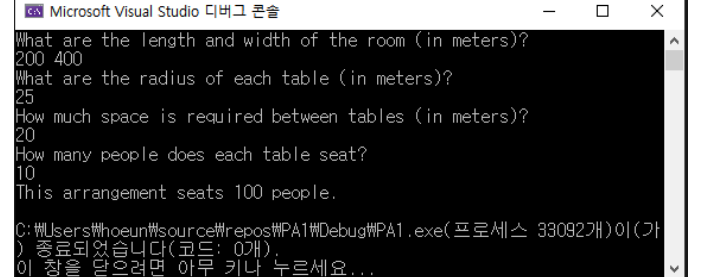
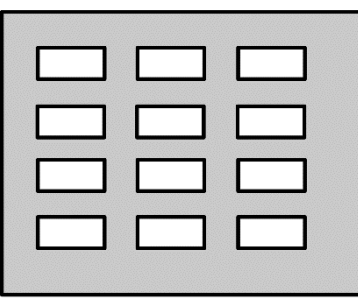
Microsoft Visual Studio 디버그 콘솔
What are the length and width of the room (in meters)?
50 30
What are the radius of each table (in meters)?
4
How much space is required between tables (in meters)?
3
How many people does each table seat?
10
This arrangement seats 80 people.

C:\Users\hwoeun\source\repos\PA1\Debug\PA1.exe(프로세스 26896개)이(가)
) 종료되었습니다(코드: 0개).
이 창을 닫으려면 아무 키나 누르세요...
  
```

- 데이터 셋을 통한 프로그램 검증

데이터 설정	
1	int roomLen = 1000, roomWid = 1000 int tableRad = 1 int interval = 1, person = 20
	프로그램 출력 결과
	
	2,217,780명
검증	
(최대의 경우)  <div> <p>1000m x 1000m의 방 안에 1m x 1m의 테이블이 1m 간격으로 배치되면 이와 같은 모습이다.</p> <p>가로로 테이블은 간격, 테이블, 간격, 테이블, ..., 테이블, 간격 순으로 놓이게 된다. 1000m일 때 1m의 간격을 두고 지름 2m 길이의 테이블은 332개가 들어갈 수 있다. (998 / 3)</p> <p>이때 332개를 배치하고 남은 공간이 2m가 남으므로 테이블이 1줄씩 더 배치될 수 있다. 즉 332 + 1개씩이다.</p> <p>가로 333개 x 세로 333개 = 110,889개이다.</p> <p>110,889개 x 20명 = 2,217,280명이다.</p> </div>	
데이터 설정	
2	int roomLen = 100, roomWid = 100 int tableRad = 40 int interval = 10, person = 1
	프로그램 출력 결과

		1명
	검증	
	(정확히 일치하는 경우)  <p>100m x 100m의 방 안에 80m x 80m 테이블이 10m 간격으로 배치되면 이와 같은 모습이다.</p> <p>실제로 테이블이 배치될 수 있는 공간의 크기는 가로, 세로 길이가 <math>100\text{m} - 2 * 10\text{m} = 80\text{m}</math>이다.</p> <p>80m x 80m의 공간 안에 80m x 80m 크기의 1개 테이블이 들어갈 수 있다. (정확히 일치)</p> <p>1개의 테이블에 1명이 앉는다면 1명이다.</p>	
3	데이터 설정	
	<pre>int roomLen = 100, roomWid = 100 int tableRad = 110 int interval = 1, person = 1</pre>	
	프로그램 출력 결과	
		0명
	검증	
	(초과의 경우) 100m x 100m 방에 220m x 220m 크기의 테이블이 들어갈 수 없으므로 0명이다.	
4	데이터 설정	
	<pre>int roomLen = 100, roomWid = 100 int tableRad = 25 int interval = 15, person = 1</pre>	
	프로그램 출력 결과	
		1명
	검증	

		<p>100m x 100m의 방 안에 50m x 50m 테이블이 15m 간격으로 배치되면 이와 같은 모습이다.</p> <p>실제로 테이블이 배치될 수 있는 공간의 크기는 가로, 세로 길이가 <math>100\text{m} - 2 * 15\text{m} = 70\text{m}</math>이다.</p> <p>70m x 70m의 공간 안에 50m x 50m 크기의 1개 테이블이 들어갈 수 있다. (남은 공간의 가로, 세로 길이는 20m 이므로 테이블이 들어갈 수 없다.)</p> <p>1개의 테이블에 1명이 앉는다면 1명이다.</p>
5	데이터 설정	
	<pre>int roomLen = 200, roomWid = 200 int tableRad = 25 int interval = 20, person = 10</pre>	
	프로그램 출력 결과	
		100명
	검증	
		<p>가로의 경우 <math>160\text{m}(200\text{m} - 2*20\text{m})</math> 안에 지름 50m 길이의 테이블이 20m의 간격을 두고 2개가 들어갈 수 있다. <math>(160 / (50 + 20))</math> 나머지는 20m이므로 50m의 테이블이 들어갈 수 없다.</p> <p>세로의 경우 <math>360\text{m}(200\text{m} - 2*20\text{m})</math> 안에 지름 50m 길이의 테이블이 20m의 간격을 두고 5개가 들어갈 수 있다. <math>(360 / (50 + 20))</math> 나머지는 10m이므로 50m의 테이블이 들어갈 수 없다.</p> <p>가로 2개 x 세로 5개 x 테이블 당 10명 = 100명이다.</p>

## 6) 토의 사항

- 비슷한 유형의 문제는 알고리즘을 잘 개발한다면 여러 경우에 변형하여 사용할 수 있다는 것을 배웠다.
- 1번 문제와 알고리즘이 거의 동일하다. 문제가 요구하는 것이 같다면 알고리즘도 같을 수 있다.
- 이 경우도 많은 코드 수정과 디버깅을 통해 코드를 완성하였다. 수학적 원리가 적용되는 코드는 알고리즘 개발이 매우 중요하다는 것을 배웠다.

### Question 3. 기차 충돌 시간 계산 문제

#### 1) 문제 정의

양방향의 철로에서 두 대의 기차가 마주보며 달려오고 있다. 두 기차 사이의 거리, 각 기차의 이동속도를 입력 받고, 충돌까지 남은 총 시간을 분과 시, 분, 초 단위 두 가지로 출력하고, 각 기차의 이동 거리를 출력한다.

입력 조건	출력 조건
<ul style="list-style-type: none"> <li>- 기차간 거리는 킬로미터 단위(km)로 양의 정수이다.</li> <li>- 기차의 이동 속도는 시간당 킬로미터 단위 (km/h)로 양의 정수이다.</li> <li>- 변수 선언은 입력 조건에 맞는 자료형을 사용한다.</li> <li>- 사용자의 입력 값이 입력 조건을 만족하는지 검사할 필요는 없으며, 입력 조건에 맞는 입력 값이 주어진다고 가정한다.</li> </ul>	<ul style="list-style-type: none"> <li>- 입력 값에 따라, 충돌까지의 시간과 각 기차의 이동 거리를 출력한다.</li> <li>- 분 단위의 시간은 소수점 이하 3자리까지만 출력한다.</li> <li>- 시, 분, 초 단위의 시간은 모두 정수이다.</li> <li>- 이동 거리는 킬로미터 단위로 소수점 이하 2자리까지만 출력한다.</li> </ul>

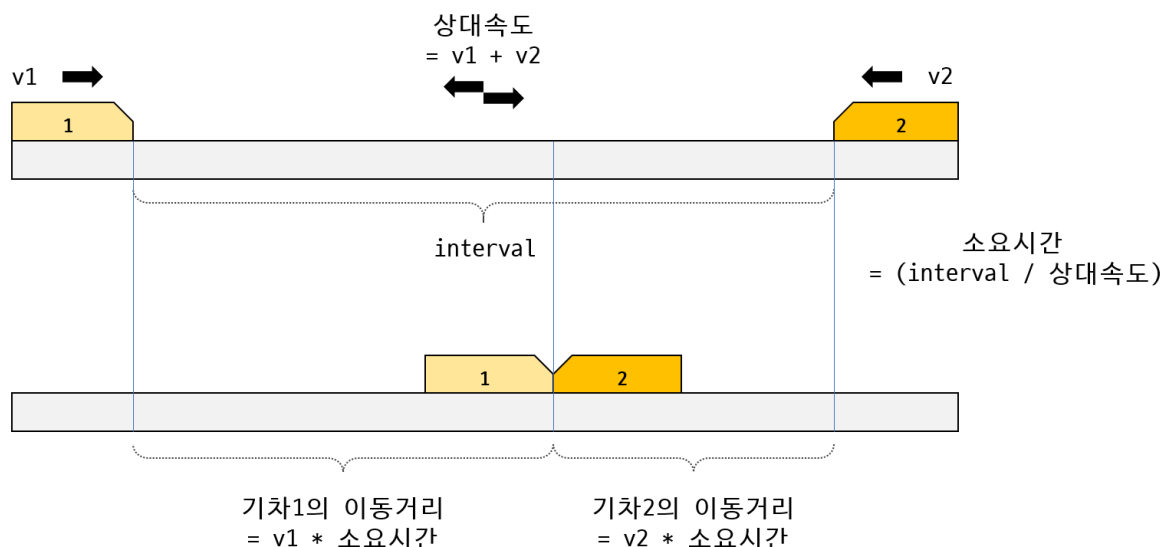
#### 2) 주요 변수 설명

변수명	자료형	설명
<b>interval</b>	int	두 기차 사이의 간격을 저장한다. (km)
<b>v1</b>	int	첫 번째 기차의 이동속도를 저장한다. (km/h)
<b>v2</b>	int	두 번째 기차의 이동속도를 저장한다. (km/h)
<b>t</b>	double	충돌까지의 소요 시간을 저장한다. (초 단위)
<b>h</b>	double	충돌까지의 소요 시간(t)을 시, 분, 초로 변경하였을 때의 시간을 저장한다. (hour)
<b>m</b>	unsigned char	충돌까지의 소요 시간(t)을 시, 분, 초로 변경하였을 때의 분을 저장한다. (min)
<b>s</b>	unsigned char	충돌까지의 소요 시간(t)을 시, 분, 초로 변경하였을 때의 초를 저장한다. (sec)
<b>MINTOSEC</b>	const int	1분을 초 단위로 변경하였을 때의 값을 저장한다.
<b>HOURTOSEC</b>	const int	1시간을 초 단위로 변경하였을 때의 값을 저장한다.

- MINTOSEC의 값인 60과 HOURTOSEC의 값인 3600은 정해진 값이므로 기호상수로 정의해주었다.

- 0~60의 범위 내 정수를 값으로 가지는 m(분 단위)과 s(초 단위) 변수는 unsigned char 타입으로 선언해주었다.

#### 3) 아이디어 및 알고리즘



- 두 대상이 서로 반대 방향으로 이동하고 있을 때, 전체 거리를 이동하는데 소요한 시간을 구하고자 한다면 두 대상의 상대속도를 구한 후, 거리를 상대속도로 나누어야 한다. 상대속도는 두 기차가 반대방향으로 이동하므로  $v1 - (-v2) = v1 + v2$ 이며, 거리는 두 기차 사이의 거리이다. 즉, 총 소요시간은 (거리/속도)인  $(interval / (v1 + v2))$ 가 된다.

- 세 변수에는 시간단위인 km/h, 그리고 이에 상응하는 km 단위의 값이 들어가 있기 때문에 연산을 할 경우 소요 시간의 단위는 시가 된다. 이때,  $(interval / (v1 + v2))$ 는 정수형 변수 피연산자들로 구성된 연산식이므로 결과도 정수로 출력된다. 이렇게 될 경우 시 단위의 소요 시간에서 소수점 이하가 절삭된다면 분, 초로 변환하였을 때 절삭오류가 발생할 수 있다. 즉, 세 변수를 double형으로 캐스팅하여 결과를 double로 반환해주어야 한다.

- 위 연산에서 계산한 값은 시 단위의 값이므로 가장 작은 단위인 초 단위로 변환해주기 위해 HOURTOSEC(3600)을 곱한다. (작은 단위로 계산해야 나눗셈과 모듈러 연산(mod)을 통해 큰 단위의 값으로 변환하기 쉬워진다.)

- 가장 큰 단위는 시간 단위 이므로 초 단위의 t를 HOURTOSEC으로 나누어준다. 이때, t의 자료형을 임시적으로 캐스팅 연산을 통해 정수형(int)으로 변환해준다. (캐스팅 연산은 변수 자체의 자료형을 변환하는 것이 아니라 임시적으로 변환하는 것이기 때문에 t에 저장된 값 자체가 절삭되거나 변하지는 않는다.) 피연산자를 전부 정수로 만들어 결과가 정수로 반환되게 한다. 시간은 정수값을 취하기 때문이며, 나머지를 분, 초로 표시하여야 하기 때문이다.

- 위와 같은 원리로 분, 초를 구한다. 이때, 초 단위의 소요시간인 t의 값을 HOURTOSEC으로 나눈 나머지 값이 분, 초로 변환해야 할 값이 된다. 모듈러 연산(mod)을 이용하여 나머지를 구해준 후 MINTOSEC으로 나누어 분 단위 시간을 구한다. 초 단위는 위와 마찬가지로 HOURTOSEC과 MINTOSEC으로 나눈 후의 나머지가 초 단위 시간이 된다. 이 때에도 마찬가지로, 시간은 정수값을 취하므로 형변환연산을 통해 피연산자를 정수형으로 캐스팅하여 결과가 정수형으로 반환되도록 한다.

- 각 기차의 이동거리는 (거리 = 속력 x 시간)이므로 각 기차의 속력(v1과 v2)의 총 소요시간을 곱해주면 각 기차의 이동거리를 계산할 수 있다. 이 때, 거리와 각 기차의 속도는 정수형 변수이고 시간은 double형이기 때문에 결과는 double형으로 반환된다. 이때, t는 초단위의 시간이 저장되어 있으므로 HOURTOSEC으로 나누어 시간단위로 변환해주어야 한다.

- 이동 거리는 소수점 이하 2자리, 분 단위의 시간은 소수점 이하 3자리로 출력되므로 %.2lf, %.3lf와 같은 적절한 형식지정자를 이용한다.

- 알고리즘

- ㉠ 각 변수의 값을 scanf() 함수를 이용해 입력 받는다.
- ㉡ 초 단위의 소요시간을 구한다. (거리 / 상대속도 \* 시간 당 초)
- ㉢ 초 단위의 소요시간을 위에 서술한 연산을 통해 시, 분, 초로 변환하여 각각의 변수에 값을 대입한다.
- ㉣ 분 단위의 소요시간을 소수점 이하 3째자리까지 출력한다. 또한, 시, 분, 초로 변환한 시간을 출력한다.
- ㉤ 각 기차의 이동거리를 출력한다. (소요시간(시간 단위) \* 각 기차의 속도)
- ㉥ 변수에 저장된 총 사람 수를 printf() 함수를 이용해 출력한다.
- ㉦ main 함수를 종료한다. (return 0)

#### 4) 소스코드

```
#include <stdio.h>

/* Programming Assignment I *
 * 0451분반 202011353 이호은 */

int main(void) {

    /* 변수 선언 */
    int interval;           // 두 기차 사이의 거리(km)
    int v1, v2;             // 각 기차의 이동속도(km/h)
    double t, h;            // 시 단위, 소요 시간
    unsigned char m, s;     // 분, 초 단위
    const int MINTOSEC = 60;
    const int HOURTOSEC = 3600;

    /* 데이터 입력 */
    printf("두 기차 사이의 거리를 입력하시오: \n");
    scanf("%d", &interval);
    printf("첫 번째 기차의 속도 (km/h)를 입력하시오: \n");
    scanf("%d", &v1);
    printf("두 번째 기차의 속도 (km/h)를 입력하시오: \n");
    scanf("%d", &v2);

    /* 소요 시간(초) = 거리 / 상대속도 * (시간 당 초) */
    t = (double)interval / ((double)v1 + (double)v2) * HOURTOSEC;

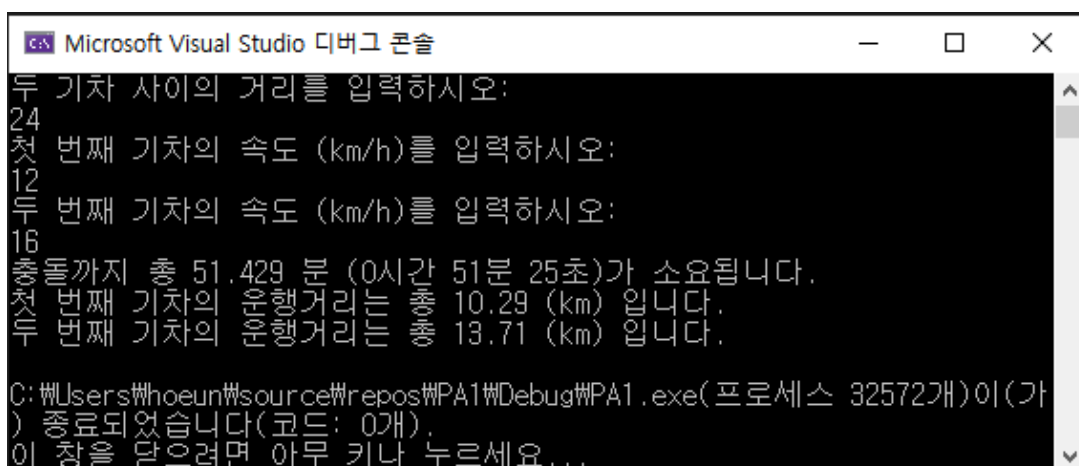
    /* 초 단위의 소요 시간을 시, 분, 초로 변환 */
    h = (int)t / HOURTOSEC;
    m = (int)t % HOURTOSEC / MINTOSEC;
    s = (int)t % HOURTOSEC % MINTOSEC;

    /* 결과 출력 */
    printf("충돌까지 총 %.31f 분 (%d시간 %d분 %d초)가 소요됩니다.\n", t / MINTOSEC, (int)h, m, s);
    printf("첫 번째 기차의 운행거리는 총 %.21f (km) 입니다.\n", t / HOURTOSEC * v1);
    printf("두 번째 기차의 운행거리는 총 %.21f (km) 입니다.\n", t / HOURTOSEC * v2);

    return 0;
}
```

#### 5) 수행 결과

- 주어진 Sample Data를 이용한 수행



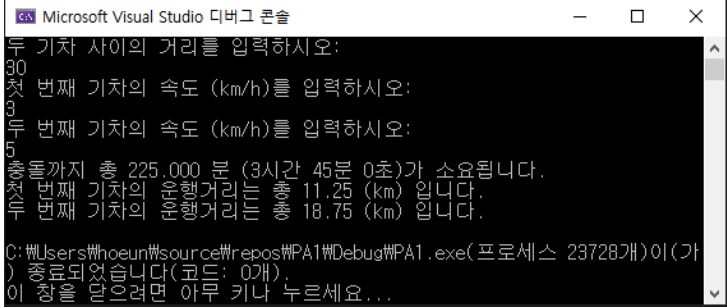
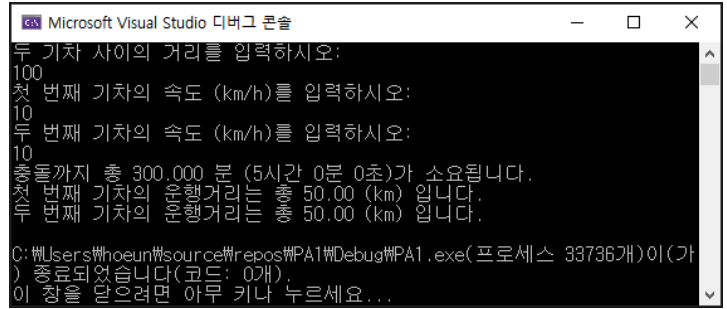
The screenshot shows the Microsoft Visual Studio Debug Console window. The title bar reads "Microsoft Visual Studio 디버그 콘솔". The console output is as follows:

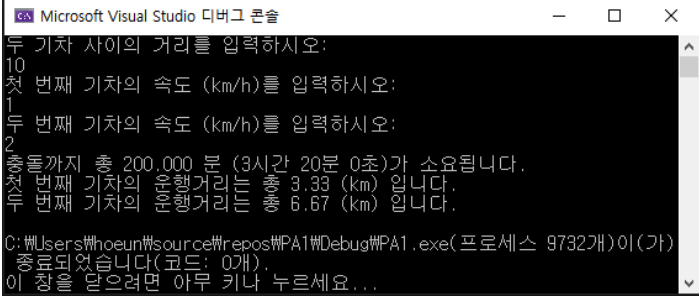
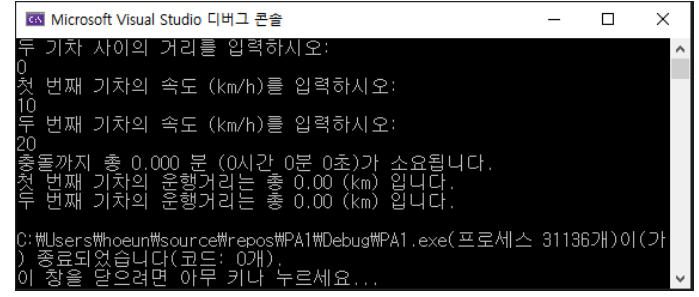
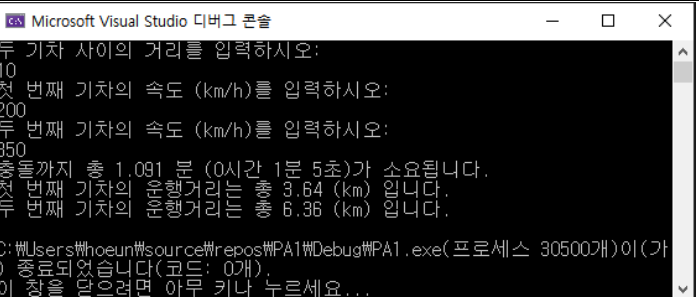
```
두 기차 사이의 거리를 입력하시오:
24
첫 번째 기차의 속도 (km/h)를 입력하시오:
12
두 번째 기차의 속도 (km/h)를 입력하시오:
16
충돌까지 총 51.429 분 (0시간 51분 25초)가 소요됩니다.
첫 번째 기차의 운행거리는 총 10.29 (km) 입니다.
두 번째 기차의 운행거리는 총 13.71 (km) 입니다.

C:\Users\hwoeun\source\repos\PA1\Debug\PA1.exe(프로세스 32572개)이(가)
) 종료되었습니다(코드: 0개).
이 창을 닫으려면 아무 키나 누르세요...
```



- 데이터 셋을 통한 프로그램 검증

1	데이터 설정	
	<pre>int interval = 30 int v1 = 3 int v2 = 5</pre>	
	프로그램 출력 결과	
		<p>225분, 3시간 45분 0초 소요                      운행거리 각각 11.25km, 18.75km</p>
2	검증	
	<p>두 기차의 상대속도는 <math>3\text{km/h} + 5\text{km/h} = 8\text{km/h}</math>이다. 시간 = 거리 / 속력이므로 <math>30\text{km} / 8\text{km/h} = 3.75\text{h}</math> 즉, 3.75시간이 소요된다. 3.75시간은 225분이다. (<math>3.75 \times 60\text{분}</math>) 그리고, 225분은 3시간 45분 0초이다.</p> <p>거리 = 속력 <math>\times</math> 시간이므로 기차1은 <math>3.75\text{h} \times 3\text{km/h} = 11.25\text{km}</math>를 이동하며                      기차2는 <math>3.75\text{h} \times 5\text{km/h} = 18.75\text{km}</math>를 이동한다.</p>	
	데이터 설정	
	<pre>int interval = 100 int v1 = 10 int v2 = 10</pre>	
	프로그램 출력 결과	
		<p>300분, 5시간 0분 0초 소요                      운행거리 각각 50km, 50km</p>
3	검증	
	<p>두 기차의 상대속도는 <math>10\text{km/h} + 10\text{km/h} = 20\text{km/h}</math>이다. 시간 = 거리 / 속력이므로 <math>100\text{km} / 20\text{km/h} = 5\text{h}</math> 즉, 5시간이 소요된다. 5시간은 300분이다. (<math>5 \times 60\text{분}</math>) 그리고, 300분은 5시간 0분 0초이다.</p> <p>거리 = 속력 <math>\times</math> 시간이므로 기차1은 <math>5\text{h} \times 10\text{km/h} = 50\text{km}</math>를 이동하며                      기차2는 <math>5\text{h} \times 10\text{km/h} = 50\text{km}</math>를 이동한다.</p>	
	데이터 설정	
	<pre>int interval = 10 int v1 = 1 int v2 = 2</pre>	
	프로그램 출력 결과	

		<p>200분, 3시간 20분 0초 소요          운행거리 각각 3.33km, 6.67km</p>
	<p>검증</p> <p>두 기차의 상대속도는 <math>1\text{km/h} + 2\text{km/h} = 3\text{km/h}</math>이다. 시간 = 거리 / 속력이므로 <math>10\text{km} / 3\text{km/h} = 3.33\text{h}</math> 즉, 3.33시간이 소요된다. 3.33시간은 200분이다. (<math>3.33 \times 60\text{분}</math>) 그리고, 200분은 3시간 20분 0초이다.          거리 = 속력 x 시간이므로 기차1은 <math>3.33\text{h} \times 1\text{km/h} = 3.33\text{km}</math>를 이동하며          기차2는 <math>3.33\text{h} \times 2\text{km/h} = 6.67\text{km}</math>를 이동한다. (소수점 셋째자리에서 반올림하였다.)</p>	
4	<p>데이터 설정</p> <p>int interval = 0          int v1 = 10          int v2 = 20</p>	
	<p>프로그램 출력 결과</p>	
		<p>0분, 0시간 0분 0초 소요          운행거리 각각 0km, 0km</p>
	<p>검증</p> <p>두 기차의 상대속도는 <math>10\text{km/h} + 20\text{km/h} = 30\text{km/h}</math>이다. 시간 = 거리 / 속력이므로 <math>0\text{km} / 30\text{km/h} = 0\text{h}</math> 즉, 거리가 0km라면 충돌하는 순간이다. 0시간은 0분이다.          거리 = 속력 x 시간이므로 기차1은 <math>0\text{h} \times 0\text{km/h} = 0\text{km}</math>를 이동하며          기차2는 <math>0\text{h} \times 0\text{km/h} = 0\text{km}</math>를 이동한다. 즉, 충돌하는 순간인 경우이다.</p>	
5	<p>데이터 설정</p> <p>int interval = 10          int v1 = 200          int v2 = 350</p>	
	<p>프로그램 출력 결과</p>	
		<p>1.091분, 0시간 1분 5초 소요          운행거리 각각 3.64km, 6.36km</p>
	<p>검증</p> <p>두 기차의 상대속도는 <math>200\text{km/h} + 350\text{km/h} = 550\text{km/h}</math>이다.          시간 = 거리 / 속력이므로 <math>10\text{km} / 550\text{km/h} = 0.018\text{h}</math></p>	

즉, 0.018시간이 소요된다. 0.018시간은 1.091분이다. (0.018 x 60분) 그리고, 1.091분은 0시간 1분 5초이다. 거리 = 속력 x 시간이므로 기차1은 0.018h x 200km/h = 3.64km를 이동하며 기차2는 0.018h x 350km/h = 6.36km를 이동한다. (소수점 셋째 자리에서 반올림하였다.)
---

#### 6) 토의 사항

- 이 프로그램은 물리학 지식이 필요하다. 두 물체의 이동거리와 이동시간은 상대속도(반대방향이므로 절댓값의 합)를 통해 구한다는 사실이다.
- 프로그램을 짜며 수학적 원리가 기반이 된 물리학적인 문제도 수학기반의 프로그래밍으로 손쉽게 계산할 수 있다는 것
- 소수점 3째자리에서 반올림하였더니 오차가 매우 작아 무시할 수 있는 수준이었다.
- 프로그래밍에서 변수의 메모리 할당 등이 실행 속도에 영향을 미친다는 것을 배웠다.
- 범위가 크지 않은 양의 정수값은 unsigned char형 변수로 사용하면 1바이트의 메모리만 할당 받아 실행속도를 증가시키고, 불필요한 메모리 낭비를 덜 수 있다.