**틀린 부분 1)**

|  |  |
| --- | --- |
| **틀린 코드의 line number** | 53 |
| **수정 전 원래 코드 내용** | 없음 |
| **수정 후 코드 내용** | sigma = 0; |
| **코드 오류 이유 설명** | sigma값이 처음에 53부터 시작해서 더해지는데 0으로 초기화시켜서 돌려줘야 한다. 아래 캡처 부분에서 sigma가 while loop에 들어가기 전에 53이 되어있음을 알 수 있다. |
| **해당 코드 부분 GDB 화면 캡처** | |
|  | |

**틀린 부분 2)**

|  |  |
| --- | --- |
| **틀린 코드의 line number** | 54 |
| **수정 전 원래 코드 내용** | while(i < last\_number) |
| **수정 후 코드 내용** | while(i <= last\_number) |
| **코드 오류 이유 설명** | 1부터 last\_number까지(sigma 1 to last\_number)더하려면 등호가 존재해야 자기자신까지 더할 수 있다. 하지만 기존의 코드는 자기자신 전까지 더한다. |
| **해당 코드 부분 GDB 화면 캡처** | |
|  | |

**틀린 부분 3)**

|  |  |
| --- | --- |
| **틀린 코드의 line number** | 21 |
| **수정 전 원래 코드 내용** | if(a=b) |
| **수정 후 코드 내용** | if(a==b) |
| **코드 오류 이유 설명** | first number와 second number를 비교할 때, if문 안에 a==b를 넣어야 true or false를 판별해낼 수 있다. 만약 if(a=b)를 넣게 된다면 값이 다를 때에도 same이 출력되게 된다. |
| **해당 코드 부분 GDB 화면 캡처** | |
|  | |

**틀린 부분 4)**

|  |  |
| --- | --- |
| **틀린 코드의 line number** | 39 |
| **수정 전 원래 코드 내용** | 없음 |
| **수정 후 코드 내용** | break |
| **코드 오류 이유 설명** | break가 없어서 I am Absent!뒤에 fall through가 일어나 Attendence Check Error!가 출력되었다. |
| **해당 코드 부분 GDB 화면 캡처** | |
| 1. 출석이 O인 경우에는 break가 존재하여 fall through가 발생하지 않는다.     2. 출석이 X인 경우에는 fall through가 발생하여 항상 두 개의 문장이 동시에 출력된다. | |