**2023104322 현시온 LAB 1.**

**1번.**

풀이) char 자료 9개를 담을 수 있는 문자 배열로 정의한 string은 9바이트를 차지하며, int와 float는 4바이트를 일반적으로 차지한다. 한편 **4바이트 정렬 규칙**에 따라 9바이트를 차지하는 string을 메모리에 할당한 다음 추가로 **패딩** 작업이 필요하다. 즉 string 자료를 메모리에 할당할 때마다 순수 string의 크기인 9바이트만큼이 아닌 9 + 3 = 12바이트만큼 오프셋이 누적된다. 한편 int와 float 자료는 4바이트이므로 패딩 작업이 필요없다. 또한 **마지막 멤버까지 포함한 오프셋**을 곧 해당 **구조체의 총 크기**라고 볼 수 있기 때문에 구조체의 총 크기는 36바이트다. (이는 멤버의 길이를 모두 더한 결과에 4바이트 정렬 규칙을 맞추는 방식으로도 구할 수 있다.)

답)

|  |  |  |
| --- | --- | --- |
| Member | Length | Offset |
| firstName | 9 bytes | 0 bytes |
| lastName | 9 bytes | 12 bytes |
| Id | 4 bytes | 24 bytes |
| gpa | 4 bytes | 28 bytes |
| currentHours | 4 bytes | 32 bytes |
| totalHours | 4 bytes | 36 bytes |

**2번.**

풀이) Base\_Address(student) = 100, Offset(gpa) = 28. 따라서 Address(student.gpa), 즉 멤버 변수 gpa의 주소는 100 + 28 = 128 이다. **(멤버 변수의 주소 = 구조체의 기본 주소 + 멤버 변수의 오프셋)**

답) 128

**3번.**

풀이) StudentRecord 구조체 하나의 크기가 36바이트이고, students는 100개의 구조체를 포함하고 있으므로 36 \* 100 = 3600바이트만큼의 메모리 공간을 준비한다.

답) 3600 bytes