**UNION**

**1. Refer the code below and comment on size of the given structure considering**

**a. Structure as union**

**b. Structure as struct**

**c. arr**

**d. uarr**

**\_\_\_ Job**

**{**

**char name[32];**

**unsigned short ucount;**

**float salary;**

**int workerNo;**

**char \*orgname;**

**};**

**\_\_\_ Job myvar; //could of union or of struct**

**Struct Job arr[10];**

**Union Job uarr[10];**

**A:**

**a) union Job:**Size of union Job = 32 bytes (largest member size).

**b) struct Job:**Size of struct Job = 50 bytes (sum of member sizes, with padding).

**c) arr[10] (array of structures):**Size of arr[10] = 500 bytes (10 structures, each 50 bytes).

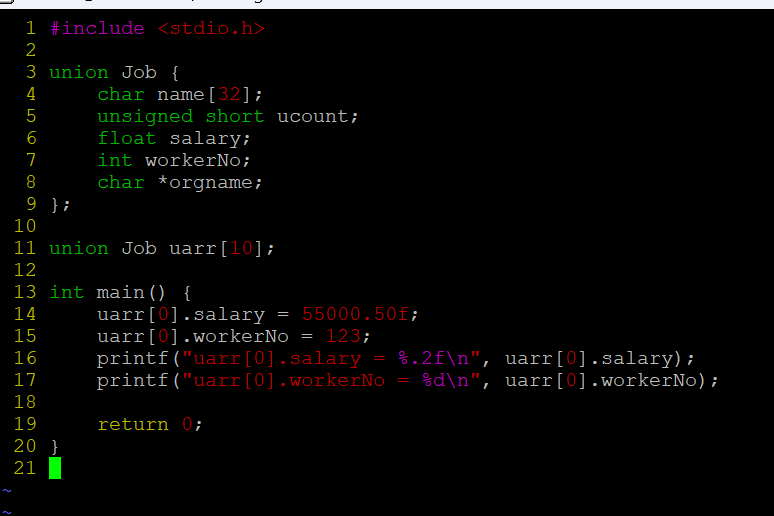
**d) uarr[10] (array of unions):**Size of uarr[10] = 320 bytes (10 unions, each 32 bytes).

**2. Refer Job datastructure in Q#1 above. Using uarr, perform below operations.**

**a. Read and store salary**

**b. Read and store workerNo**

**Comment on values of output if salary and workerNo are printed in order. Justify your statement.**

****

**OUTPUT:**

**A black background with white text

Description automatically generated**

**3. Refer Job datastructure in Q#1 above. Assume that myvar is a structure variable. If I need to place 2 bytes (i.e 0x0102) as ucount using a char \*ptr then list all possible statements that can be used in \_\_\_\_\_.**

**[Let solutions include cases such as**

**i. using base address of ucount**

**ii. using relative address of ucount w.r.t to base address of myvar]**

**int main()**

**{**

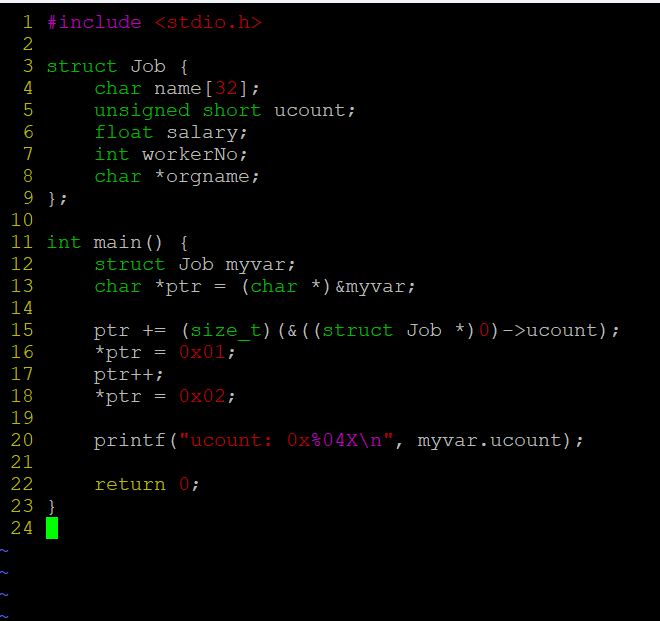
**char \*ptr = &myvar;**

**\_\_\_\_\_\_\_\_\_\_\_ = 0x01;**

**\_\_\_\_\_\_\_\_\_\_\_ = 0x02**

**}**

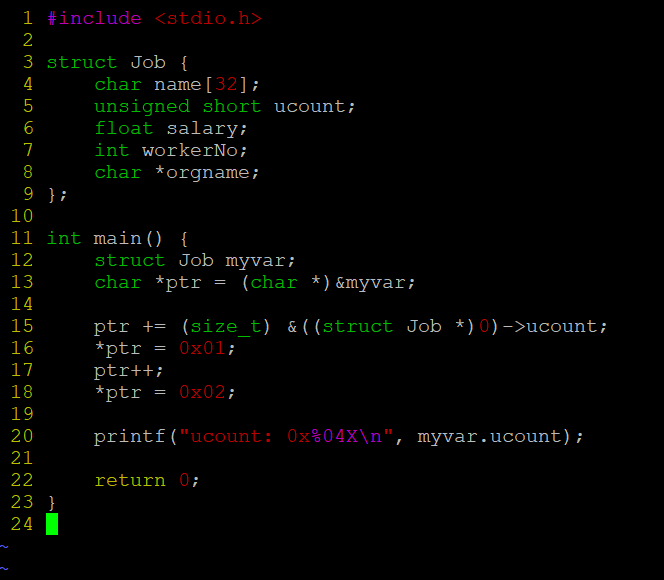
**i. using base address of ucount:**

****

**OUTPUT:**

****

**ii. using relative address of ucount w.r.t to base address of myvar]**

****

**OUTPUT:**

****