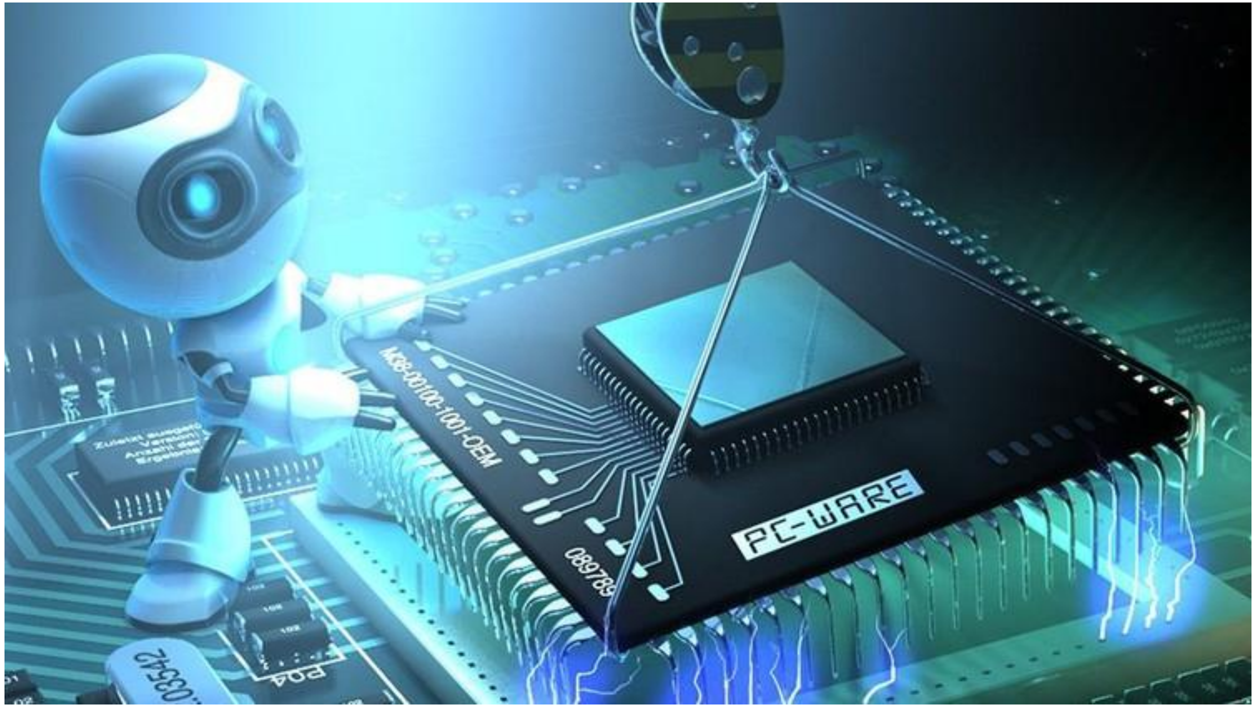
Exam 6 - Problem 2



Exam 6 - Problem 2

Given the following code in C:

Exam 6 - Problem 2

```
struct employee {
        char group;
        int emp_id;
        int age;
        char gender;
        int years_experience;
}

int salary(employee A, int deposit, int times) {
        int count = 0;
        while (count < times) {
                if (A.group == 'X') { deposit = deposit + 10; }
                else if (A.group == 'Y' { deposit = deposit + 40; }
                else { deposit = deposit + 80; }
                ++count;
                }

        return A.years_experience * deposit;
}
```

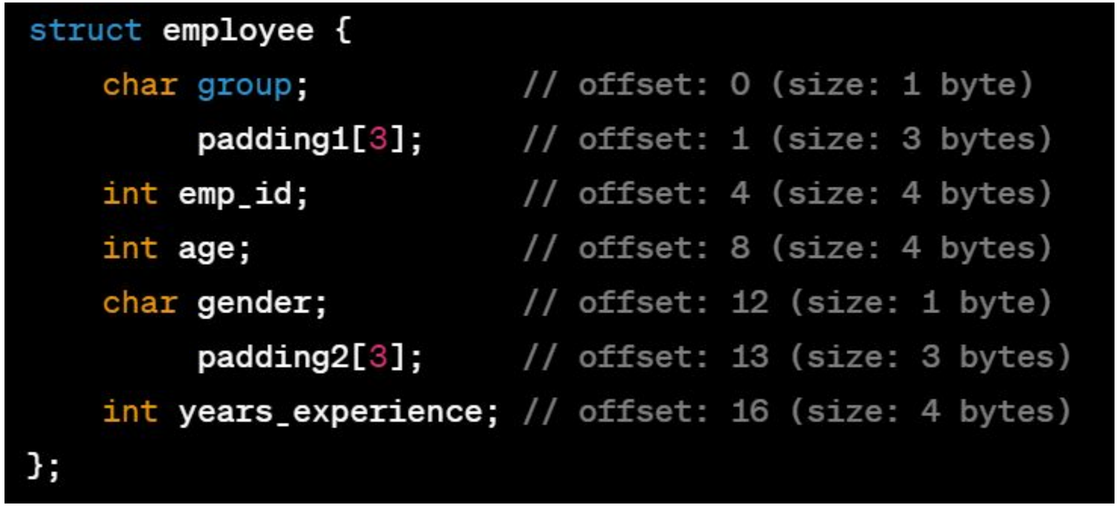
Exam 6 - Problem 2

a) Draw how the `employee` structure would be stored, clearly indicating the offsets from the start and the size of all the fields.

b) Translate the `salary` subroutine to x86 assembler.

Exam 6 - Problem 2

a) Draw how the `employee` structure would be stored, clearly indicating the offsets from the start and the size of all the fields.



```
struct employee {
    char group;               // offset: 0 (size: 1 byte)
         padding1[3];         // offset: 1 (size: 3 bytes)
    int emp_id;               // offset: 4 (size: 4 bytes)
    int age;                  // offset: 8 (size: 4 bytes)
    char gender;              // offset: 12 (size: 1 byte)
         padding2[3];         // offset: 13 (size: 3 bytes)
    int years_experience;     // offset: 16 (size: 4 bytes)
};
```

Lucas Bazilio - Udemy

Exam 6 - Problem 2

b)

```
salary:
        pushl %ebp
        movl %esp, %ebp
        subl $4, %esp
        # initialize count to 0
        movl $0, -4(%ebp)
        movl 8(%ebp), %ecx  # %ecx = employee A
loop:
        cmpl 16(%ebp), -4(%ebp)          # compare count with times
        jge endloop                      # if count >= times, exit the loop
```

```
int salary(employee A, int deposit, int times) {
        int count = 0;
        while (count < times) {
                if (A.group == 'X') { deposit = deposit + 10; }
                else if (A.group == 'Y' { deposit = deposit + 40; }
                else { deposit = deposit + 80; }
                ++count;
                }

        return A.years_experience * deposit;
}
```

Part 1/3

x86

Lucas Bazilio - Udemy

Exam 6 - Problem 2

b)

Part 2/3

```
        movzx %ecx, %eax            # load A.group into %eax
        cmpb $'X', %al              # compare A.group with 'X'
        je groupX                   # if A.group == 'X', jump to groupX
        cmpb $'Y', %al              # compare A.group with 'Y'
        je groupY                   # if A.group == 'Y', jump to groupY
        movl $80, %eax              # otherwise, set %eax to 80
        jmp adddeposit              # jump to adddeposit
groupX:
        movl $10, %eax              # add 10 to %eax
        jmp adddeposit              # jump to adddeposit
groupY:
        movl $40, %eax              # add 40 to %eax
```

x86

Lucas Bazilio - Udemy

Exam 6 - Problem 2

b)

Part 3/3

```
adddeposit:
        addl %eax, 12(%ebp)          # add %eax to deposit
        addl $1, -4(%ebp)            # increment count by 1
        jmp loop                     # jump back to the loop
endloop:
        movl 16(%ecx), %eax                # load A.years_experience into %eax
        imul 12(%ebp), %eax                # multiply by deposit
        addl $4, %esp
        mov %ebp, %esp                     # deallocate local variables
        pop %ebp
        ret
```

x86