Memory

Introduction

This is a variation on the last lab where the task was to save XYZ Company Corp. employees information in a file. The goal here is to store employee information using C++ structures. In order to store multiple employees we will use an array of structures. The goal is get a sense of how this array is stored in memory by printing memory addresses.

```cpp
#include <iostream>
#include <ctime>
#include <string>
using namespace std;

char Alphabet[] = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
char alphabet[] = "abcdefghijklmnopqrstuvwxyz";
char digits[] = "0123456789";

// Converts a character between 0 and 9
// to an integer value
int char_to_int(char c)
{
    // TO DO
}

// Picks a random char from an array of
// characters with length n
char choose_char(char* ptr_c, int n)
{
    // TO DO
}

// Generates a random name having length between
```
// 5 and 15 letters.
// The first letter of the name is capitalized.
std::string gen_random_name(int min_len, int max_len)
{
    std::string n;

    // TO DO

    return n;
}

struct Employee {
    std::string firstname;
    std::string lastname;
    char skill; // Between 0 and 9
    int bonus; // Between 0 and 9 - tied to skill
};

const int NUM_EMPLOYEES = 5;

void diagnostics_mem_layout(Employee* employees, int n)
{
    // Task 1: Print the address for employee 0 to 4
    cout << "Address of the 0th employee: " << endl;
    cout << "Address of the 1th employee: " << endl;
    cout << "Address of the 2th employee: " << endl;
    cout << "Address of the 3th employee: " << endl;
    cout << "Address of the 4th employee: " << endl;

    // Task 2: Compute the offset between employee 2 and 3, and
    // between employee 0 and 1. Is it the same?
    cout << "Offset between employee 2 and 3: " << endl;
    cout << "Offset between employee 0 and 1: " << endl;

    // Task 3: How much space does employee 0-4 takes in the memory?
    // Use sizeof() to compute the size of the Employee struct
    cout << "Size of employee 0: " << endl;
    cout << "Size of employee 1: " << endl;
    cout << "Size of employee 2: " << endl;
    cout << "Size of employee 3: " << endl;
    cout << "Size of employee 4: " << endl;

    // Task 3 - part 2: Hmm. Does every employee takes the same amount of
    // space in the memory? Change the min and max length for names and see
    // if you get the same result. If you get the same result, explain
    // why this is so?

    // TO DO
// Task 3 - part 3: Use the offset that you computed in Task 2 above
// to confirm that the offset is similar to the size of each array entry.

// TO DO

//--------------------------------------------------------
// Task 4: Print the memory address where skill level and bonus for
// employee 3 is stored. How much space skill level takes and how
// much space bonus takes?
//
// You'll notice that you need to cast to address of skill, which is a char,
// to (void *). This is because the << operator is treating this as
// a char array.
//
// Pointers are arrays are intimately tied.

cout << "Skill for employee 3: " << endl;
cout << "Bonus for employee 3: " << endl;

int main()
{
    srand(time(NULL));

    // Array of employees
    Employee employees[NUM_EMPLOYEES];

    for (int i=0; i<NUM_EMPLOYEES; ++i) {
        employees[i].firstname = gen_random_name(5, 10);
        employees[i].lastname = gen_random_name(5, 10);
        employees[i].skill = choose_char(digits, 10);
        employees[i].bonus = char_to_int(employees[i].skill);
    }

    // Print
    for (int i=0; i<NUM_EMPLOYEES; ++i) {
        cout << "Employee " << i << endl;
        cout << "First name: " << endl;
        cout << "Last name: " << endl;
        cout << "skill: " << endl;
        cout << "bonus: " << endl;
    }

    // Call the following function to complete the following tasks
    diagnostics_mem_layout(employees, NUM_EMPLOYEES);

    return 0;
}

Submission

Please submit lab4-mem.cpp