

Practice Questions 2 (Structures and Enums)

1. Create a structure called `employeeType` that contains two members: an employee number (type `int`) and the employee's compensation (in dollars; type `float`). Ask the user to fill in this data for three employees, store it in three variables of type `struct employeeType` , and then display the information for each employee.
2. Imagine an enumerated type with values that indicate an employee type within an organization:

enum etype { laborer, secretary, manager, accountant, executive, researcher };

Write a program that first allows the user to specify a type by entering its first letter ('l' , 's' , 'm' , and so on), then stores the type chosen as a value of a variable of type `enum etype` , and finally displays the complete word for this type.

Enter employee type (first letter only) laborer, secretary, manager, accountant, executive, researcher: a
Employee type is accountant.

You'll probably need two switch statements: one for input and one for output.

3. Create a structure of type `dateType` that contains three members: the month, the day of the month, and the year, all of type `int` . (Or use day-month-year order if you prefer.) Have the user enter a date in the format 12/31/2001, store it in a variable of type `struct dateType` , then retrieve the values from the variable and print them out in the same format.
4. Add a variable of type `enum etype` (see Exercise 2), and another variable of type `struct dateType` (see Exercise 3) to the employee class of Exercise 1. Organize the resulting program so that the user enters four items of information for each of three employees: an employee number, the employee's compensation, the employee type, and the date of first employment. The program should store this information in three variables of type `employee` , and then display their contents.
5. In the heyday of the British empire, Great Britain used a monetary system based on pounds, shillings, and pence. There were 20 shillings to a pound, and 12 pence to a shilling. The notation for this old system used the pound sign, £, and two decimal points, so that, for example, £5.2.8 meant 5 pounds, 2 shillings, and 8 pence. (Pence is the plural of penny.) The new monetary system, introduced in the 1950s, consists of only pounds and pence, with 100 pence to a pound (like U.S. dollars and cents). We'll call this new system decimal pounds. Thus £5.2.8 in the old notation is £5.13 in decimal pounds (actually £5.1333333).

Create a structure called `sterling` that stores money amounts in the old-style British system. The members could be called `pounds` , `shillings` , and `pence` , all of type `int` . The program should ask the user to enter a money amount in new-style decimal pounds (type `double`), convert it to the old-style system, store it in a variable of type `struct sterling` , and then display this amount in pounds-shillings-pence format.