

**CIS-280**  
**Homework 7: Data-Directed Programming**  
**Due Tuesday April 19, 2005**  
**60 points**

This assignment will use the procedures `FindJoyAtt`, `FindSadAtt`, `FindJoyEmp`, and `FindSadEmp` that you designed in Homework-6.

The Good-Times Company purchases both Joy and Sad and is aghast to find that their personnel records are structured so differently. Since Good-Times is anticipating rapid growth as a conglomerate, they fear that other soon-to-be-acquired companies may have even different personnel file structures. Instead of redoing all the representation schemes, they decide to use data-directed programming to implement a Get-record procedure and a Get-attribute procedure that can be easily expanded should other companies with even different file organizations be purchased. Their goal is to develop a procedure **Find-Employee** that can take as arguments an employee number and a file containing both the file of joy records and the file of sad records (and files of any other companies purchased by the conglomerate) and retrieve the record for a particular employee.

To do this, each company's file structure must have manifest type— ie., its type is attached to the file. Let us assume that the type is the company's name. Then Joy's file from Homework-6 now looks as follows:

```
(JOY (((age 24)(empnum 119911))
      ((empnum 489235)(age 25)(salary 50000))
      ((salary 20000)(empnum 556647))))
```

Sad's employee file from the homework assignment now is:

```
(SAD (((age empnum salary) (25 489235 50000))
      (((empnum age)(119911 24)) () () )
      (((empnum salary)(556647 20000)) () () )))
```

When a record is retrieved from a file, it should be returned with manifest type — ie., attach the file's type to it.

1. File `~carberry/HW-7.scm` contains procedures for creating a table, putting entries into the table, and retrieving entries from it. It also contains procedures **Get-Type**, **Get-Rep**, and **Adjoin-Type** discussed in class for retrieving the type of an object with manifest type, retrieving the representation (after stripping off the type) of an object, and taking as arguments a type and the representation of an object and returning the object with manifest type.
2. Enter Scheme, load file `~carberry/HW-7.scm`, and press the Execute button on the Scheme interface. (Do not concern yourself at this time with how the procedures in my file work.)
3. Write a procedure **TableSetup** that uses `put` to form a table that will allow you to determine the appropriate procedure to call to search a file for a particular employee record or search a record for a particular attribute value. Name the columns `SAD` and `JOY`, and name the rows `Record` and `Attribute`. Once you have written procedure **TableSetup**, execute it.

- Write procedure (`Get-Attribute record att-name`) that takes a record with manifest type and an attribute name as arguments and returns the value for that attribute. Note that this procedure will use `get` to obtain the procedure (either your procedure for retrieving an attribute from a Joy record or your procedure for retrieving an attribute from a Sad record) that should be used on the record. (You may want to use a let statement to temporarily retain the procedure retrieved from the table.) For example,

```
(define r1 (Adjoin-Type 'JOY '((empnum 489235) (age 25) (salary 50000))))
(define r2 (Adjoin-Type 'SAD '((empnum age salary) (489235 25 50000))))
(Get-Attribute r1 'age) returns 25
(Get-Attribute r2 'age) returns 25
```

- Write procedure (`Get-Record file number`) that takes a file (with manifest type) and an employee number as arguments and returns the record (with manifest type) for that employee. Note that this procedure will use `get` to obtain the procedure (either your procedure for retrieving a record from the Joy file or your procedure for retrieving a record from the Sad file) that should be used on the file. (You may want to use a let statement to temporarily retain the procedure retrieved from the table. You MIGHT also want to use a let statement to retain the record extracted from the file while you test whether it is empty before adding a type to it.) For example, if `file-1` is the file

```
(JOY (((age 24)(empnum 119911))
      ((empnum 489235)(age 25)(salary 50000))
      ((salary 20000)(empnum 556647))))
```

then (`Get-Record file-1 489235`) will return the following:

```
(JOY ((empnum 489235) (age 25) (salary 50000)))
```

Similarly, if `file-2` is the file

```
(SAD (((age empnum salary) (25 489235 50000))
      (((empnum age)(119911 24)) () () )
      (((empnum salary)(556647 20000)) () () )))
```

then (`Get-Record file-2 119911`) will return the following:

```
(SAD ((empnum age)(119911 24)))
```

- Write procedure **Find-Employee** that takes a list of all the company's files (the files for JOY, the files for SAD, and the files for any other companies purchased by the conglomerate) and an employee number as arguments and returns the record for that employee. (Note that this procedure will call your procedure `Get-Record` for each of the files in the list until it finds the desired record.)
- Submit your code on the electronic submission system for Homework-7. Your submitted code must include the procedures in `~carberry/HW-7.scm`