6.1 Define a structure type auto t to represent an automobile. Include components for the make and model (strings), the odometer reading, the manufacture and purchase dates (use another user-defined type called date t), and the gas tank (use a user-defined type tank t with com ponents for tank capacity and current fuel level, giving both in gallons). Write and test I/O functions scan date, scan tank, scan auto, print date, print tank, and print auto. Here is a small data set to try:

Mercury Sable 99842 1 18 2001 5 30 1991 16 12.5

Mazda Navajo 123961 2 20 1993 6 15 1993 19.3 16.7

Problem : Automobile Data Management

Implementation:

```
#include <stdio.h>

typedef struct {
    int day;
    int month;
    int year;
} date_t;

typedef struct {
    float capacity;
    float current_level;
} tank_t;

typedef struct {
    char make[50];
    char model[50];
    int odometer;
    date_t manufacture_date;
    date_t purchase_date;
    tank_t gas_tank;
} auto_t;

void scan_date(date_t *date) {
    scanf("%d %d %d", &date->day, &date->month, &date->year);
}
```

```
void scan_tank(tank_t *tank) {
    scanf("%f %f", &tank->capacity, &tank->current_level);
}
void scan_auto(auto_t *car) {
    scanf("%s %s %d", car->make, car->model, &car->odometer);
    scan_date(&car->manufacture_date);
    scan_date(&car->purchase_date);
    scan_tank(&car->gas_tank);
}
void print_date(const date_t *date) {
    printf("%d/%d/%d", date->day, date->month, date->year);
}
void print_tank(const tank_t *tank) {
    printf("Capacity: %.1f gallons, Current Level: %.1f gallons", tank->capacity, tank->current_level);
}
void print_auto(const auto_t *car) {
    printf("Make: %s, Model: %s, Odometer: %d miles\n", car->make, car->model, car->odometer);
    printf("Manufacture Date: ");
    print_date(&car->manufacture_date);
    printf("\nPurchase Date: ");
    print_date(&car->purchase_date);
    printf("\nGas Tank: ");
    print_tank(&car->gas_tank);
    printf("\n");
}
int main() {
    auto_t cars[2];
    printf("Enter details for car 1: ");
    scan_auto(&cars[0]);
```

```
    printf("Enter details for car 2: ");
    scan_auto(&cars[1]);
    printf("\nCar 1 details:\n");
    print_auto(&cars[0]);
    printf("\nCar 2 details:\n");
    print_auto(&cars[1]);
    return 0;
}
```

Input:

Mercury Sable 99842 1 18 2001 5 30 1991 16 12.5

Mazda Navajo 123961 2 20 1993 6 15 1993 19.3 16.7

Output:

Car 1 details:

Make: Mercury, Model: Sable, Odometer: 99842 miles

Manufacture Date: 1/18/2001

Purchase Date: 5/30/1991

Gas Tank: Capacity: 16.0 gallons, Current Level: 12.5 gallons

6.2 Define a structure type element t to represent one element from the periodic table of elements. Components should include the atomic number (an integer); the name, chemical symbol, and class (strings); a numeric field for the atomic weight; and a seven-element array of integers for the number of electrons in each shell. The following are the components of an element t structure for sodium. 11 Sodium Na alkali metal 22.9898 2 8 1 0 0 0 0 Define and test I/O functions scan element and print element.

Problem: Representation and Management of Periodic Table Elements

Implementation:

```
#include <stdio.h>
typedef struct {
    int atomic_number;
    char name[30];
    char symbol[5];
```

```
    char class[30];
    float atomic_weight;
    int electrons[7];
} element_t;
void scan_element(element_t *element) {
    scanf("%d %s %s %s %f", &element->atomic_number, element->name, element->symbol, element->class, &element->atomic_weight);
    for(int i = 0; i < 7; i++) {
        scanf("%d", &element->electrons[i]);
    }
}
void print_element(const element_t *element) {
    printf("Atomic Number: %d\n", element->atomic_number);
    printf("Name: %s\n", element->name);
    printf("Symbol: %s\n", element->symbol);
    printf("Class: %s\n", element->class);
    printf("Atomic Weight: %.4f\n", element->atomic_weight);
    printf("Electrons per Shell: ");
    for(int i = 0; i < 7; i++) {
        printf("%d ", element->electrons[i]);
    }
    printf("\n");
}
int main() {
    element_t element;
    printf("Enter the element details: ");
    scan_element(&element);
    printf("\nElement details:\n");
    print_element(&element);
```

```
    return 0;
}
```

Sample Output:

Enter the element details:

11 Sodium Na alkali metal 22.9898 2 8 1 0 0 0 0

Element details:

Atomic Number: 11

Name: Sodium

Symbol: Na

Class: alkali metal

Atomic Weight: 22.9898

Electrons per Shell: 2 8 1 0 0 0 0