STRUCTURES

Practice questions:

1. student
2. customer
3. employee

1. (**student**) You are required to write a C program to process an array of student records. For each student record, it stores the student `id` and `name`. In the program, you need to write the following three functions:
   - The function `inputStud()` reads in students' information according to an input student size.
   - The function `printStud()` prints the student information on the display. It will print the message "`Empty array`" if the student list is empty.
   - The function `removeStud()` takes in three parameters. It removes the `target` student name from the array which has `*size` numbers in it. If `*size` is equal to zero the function should issue an error message "`Array is empty`". If the *`target`* name does not appear in the array, the function should issue an error message "`The target does not exist`". The program defines the constant *SIZE* as the maximum number of student records which can be stored in the array. The function will return 0 if the removal operation is successful, 1 if the array is empty or 2 if the number does not exist in the array.

   The prototypes of the three functions are given below:

```
void inputStud(Student *s, int size);
void printStud(Student *s, int size);
int removeStud(Student *s, int *size, char *target);
```

   The structure definition for the structure `Student` is given below:

```
typedef struct {
   int id;
   char name[50];
} Student;
```

   A sample program template is given below to test the functions:

```
#include <stdio.h>
#include <string.h>
#define SIZE 50
typedef struct {
   int id;
   char name[50];
} Student;
void inputStud(Student *s, int size);
void printStud(Student *s, int size);
int removeStud(Student *s, int *size, char *target);
int main()
{
   Student s[SIZE];
   int size=0, choice;
   char target[80], *p;
   int result;

   printf("Select one of the following options: \n");
   printf("1: inputStud()\n");
   printf("2: removeStud()\n");
   printf("3: printStud()\n");
   printf("4: exit()\n");
   do {
      printf("Enter your choice: \n");
```

```
            scanf("%d", &choice);
            switch (choice) {
               case 1:
                  printf("Enter size: \n");
                  scanf("%d", &size);
                  printf("Enter %d students: \n", size);
                  inputStud(s, size);
                  break;
               case 2:
                  printf("Enter name to be removed: \n");
                  scanf("\n");
                  fgets(target, 80, stdin);
                  if (p=strchr(target,'\n')) *p = '\0';
                  printf("removeStud(): ");
                  result = removeStud(s, &size, target);
                  if (result == 0)
                     printf("Successfully removed\n");
                  else if (result == 1)
                     printf("Array is empty\n");
                  else if (result == 2)
                     printf("The target does not exist\n");
                  else
                     printf("An error has occurred\n");
                  break;
               case 3:
                  printStud(s, size);
                  break;
            }
         } while (choice < 4);
         return 0;
      }
      void inputStud(Student *s, int size)
      {
          /* Write your code here */
      }
      void printStud(Student *s, int size)
      {
          /* Write your code here */
      }
      int removeStud(Student *s, int *size, char *target)
      {
          /* Write your code here */
      }
```

Some sample input and output sessions are given below:

(1) Test Case 1:

```
Select one of the following options:
1: inputStud()
2: removeStud()
3: printStud()
4: exit()
Enter your choice:
1
Enter size:
3
Enter 3 students:
Student ID:
11
Student Name:
Hui Siu Cheung
```

```
Student ID:
12
Student Name:
Kenny B
Student ID:
13
Student Name:
Victor Leong
Enter your choice:
3
The current student list:
Student ID: 11 Student Name: Hui Siu Cheung
Student ID: 12 Student Name: Kenny B
Student ID: 13 Student Name: Victor Leong
Enter your choice:
4
```

(2) Test Case 2:

```
Select one of the following options:
1: inputStud()
2: removeStud()
3: printStud()
4: exit()
Enter your choice:
1
Enter size:
3
Enter 3 students:
Student ID:
11
Student Name:
Hui Siu Cheung
Student ID:
12
Student Name:
Kenny B
Student ID:
13
Student Name:
Victor Leong
Enter your choice:
2
Enter name to be removed:
Victor Leong
removeStud(): Successfully removed
Enter your choice:
3
The current student list:
Student ID: 11 Student Name: Hui Siu Cheung
Student ID: 12 Student Name: Kenny B
Enter your choice:
4
```

(3) Test Case 3:

```
Select one of the following options:
1: inputStud()
2: removeStud()
3: printStud()
4: exit()
Enter your choice:
```

```
1
Enter size:
3
Enter 3 students:
Student ID:
11
Student Name:
Hui Siu Cheung
Student ID:
12
Student Name:
Kenny B
Student ID:
13
Student Name:
Victor Leong
Enter your choice:
2
Enter name to be removed:
Victor Hui
removeStud(): The target does not exist
Enter your choice:
3
The current student list:
Student ID: 11 Student Name: Hui Siu Cheung
Student ID: 12 Student Name: Kenny B
Student ID: 13 Student Name: Victor Leong
Enter your choice:
4
```

(4) Test Case 4:

```
Select one of the following options:
1: inputStud()
2: removeStud()
3: printStud()
4: exit()
Enter your choice:
2
Enter name to be removed:
Victor Hui
removeStud(): Array is empty
Enter your choice:
3
The current student list:
Empty array
Enter your choice:
4
```

2. (**customer**) Write a C program that repeatedly reads in customer data from the user and prints the customer data on the screen until the customer name `"End Customer"` (i.e., first_name last_name) is read. Your program should include the following two functions: the function `nextCustomer()` reads and returns a record for a single customer to the caller via a pointer parameter ***`acct`***, and the function `printCustomer()` takes a parameter ***`acct`*** and then prints the customer information. The prototypes of the two functions are given below:

```
void nextCustomer(struct account *acct);
void printCustomer(struct account acct);
```

The structure definition for **`struct account`** is given below:

```
struct account {
   struct
   {
      char lastName[10];
      char firstName[10];
   } names;
   int accountNum;
   double balance;
};
```

A sample program template is given below to test the functions:

```
#include <stdio.h>
#include <string.h>
struct account {
   struct
   {
      char lastName[10];
      char firstName[10];
   } names;
   int accountNum;
   double balance;
};
void nextCustomer(struct account *acct);
void printCustomer(struct account acct);
int main()
{
   struct account record;
   int flag = 0;

   do {
      nextCustomer(&record);
      if ((strcmp(record.names.firstName, "End") == 0) &&
          (strcmp(record.names.lastName, "Customer") == 0))
         flag = 1;
      if (flag != 1)
         printCustomer(record);
   } while (flag != 1);
}
void nextCustomer(struct account *acct)
{
   /* Write your code here */
}
void  printCustomer(struct account acct)
{
   /* Write your code here */
}
```

Some sample input and output sessions are given below:

(1) Test Case 1:

```
Enter names (firstName lastName):
SC Hui
Enter account number:
123
Enter balance:
6789.89
Customer record:
SC Hui 123 6789.89
Enter names (firstName lastName):
End Customer
```

(2) Test Case 2:

```
Enter names (firstName lastName):
SC Hui
Enter account number:
123
Enter balance:
6789.89
Customer record:
SC Hui 123 6789.89
Enter names (firstName lastName):
FY Tan
Enter account number:
13
Enter balance:
69.89
Customer record:
FY Tan 13 69.89
Enter names (firstName lastName):
End Customer
```

(3) Test Case 3:

```
Enter names (firstName lastName):
End Customer
```

3. (**employee**) Write a C program that creates an array of structures to hold the employee information below:

```
typedef struct {
   char name[40];
   char telno[40];
   int id;
   double salary;
} Employee;
```

You are required to implement the following three functions:

- The function `readin()` reads a number of persons' names and their corresponding telephone numbers, passes the data to the caller via the parameter *p*, and returns the number of names that have entered. The character `'#'` is used to indicate the end of user input.
- The function `search()` allows the user to query the array using the name field. If the name is found, the program displays the message `"Employee found at index location: x"`. The function `search()` finds the employee data of an input name `target`, and then prints the name, telephone number, id and salary on the screen. If the input name cannot be found, then it will print the error message `"Name not found"` on the screen.
- If the name is not found and the array is not full, the user can then add the name as a new record into the array. Assume that the maximum size of the array is 100. The function `addEmployee()` adds a new employee record into the array. The message `"Added at position: x"` will be printed. If the database is fill, an error message `"Database is full"` will be printed on the screen. The function returns the updated size of the array after adding the new employee record.

The prototypes of the functions are given below:

```
int readin(Employee *emp);
int search(Employee *emp, int size, char *target);
int addEmployee(Employee *emp, int size, char *target);
```

A sample program template is given below to test the functions:

```c
#include <stdio.h>
#include <string.h>
#define MAX 100
typedef struct {
   char name[40];
   char telno[40];
   int id;
   double salary;
} Employee;
void printEmp(Employee *emp, int size);
int readin(Employee *emp);
int search(Employee *emp, int size, char *target);
int addEmployee(Employee *emp, int size, char *target);
int main()
{
   Employee emp[MAX];
   char name[40], *p;
   int size, choice, result;

   printf("Select one of the following options: \n");
   printf("1: readin() \n");
   printf("2: search() \n");
   printf("3: addEmployee() \n");
   printf("4: print() \n");
   printf("5: exit() \n");
   do {
      printf("Enter your choice: \n");
      scanf("%d", &choice);
      switch (choice) {
         case 1:
            size = readin(emp);
            break;
         case 2:
            printf("Enter search name: \n");
            scanf("\n");
            fgets(name, 40, stdin);
            if (p=strchr(name,'\n')) *p = '\0';
            result = search(emp,size,name);
            if (result != 1)
               printf ("Name not found! \n");
            break;
         case 3:
            printf("Enter new name: \n");
            scanf("\n");
            fgets(name, 40, stdin);
            if (p=strchr(name,'\n')) *p = '\0';
            result = search(emp,size,name);
            if (result != 1)
               size = addEmployee(emp, size, name);
            else
               printf("The new name has already existed in the database\n");
            break;
         case 4:
            printEmp(emp, size);
            break;
         default:
            break;
      }
   } while (choice < 5);
   return 0;
}
```

```
void printEmp(Employee *emp, int size)
{
   int i;

   printf("The current employee list: \n");
   if (size==0)
      printf("Empty array\n");
   else
   {
      for (i=0; i<size; i++)
         printf("%s %s %d %.2f\n",emp[i].name,emp[i].telno,emp[i].id,
                        emp[i].salary);
   }
}
int readin(Employee *emp)
{
   /* Write your code here */
}
int search(Employee *emp, int size, char *target)
{
   /* Write your code here */
}
int addEmployee(Employee *emp, int size, char *target)
{
   /* Write your code here */
}
```

Some sample input and output sessions are given below:

(1) Test Case 1:

```
Select one of the following options:
1: readin()
2: search()
3: addEmployee()
4: print()
5: exit()
Enter your choice:
1
Enter name:
Hui Siu Cheung
Enter tel:
12345678
Enter id:
11
Enter salary:
123.45
Enter name:
#
Enter your choice:
4
The current employee list:
Hui Siu Cheung 12345678 11 123.45
Enter your choice:
5
```

(2) Test Case 2:

```
Select one of the following options:
1: readin()
2: search()
3: addEmployee()
4: print()
```

```
5: exit()
Enter your choice:
1
Enter name:
Hui Siu Cheung
Enter tel:
12345678
Enter id:
11
Enter salary:
123.45
Enter name:
Kenny B
Enter tel:
23456789
Enter id:
12
Enter salary:
1234.45
Enter name:
#
Enter your choice:
4
The current employee list:
Hui Siu Cheung 12345678 11 123.45
Kenny B 23456789 12 1234.45
Enter your choice:
2
Enter search name:
Kenny B
Employee found at index location: 1
Kenny B 23456789 12 1234.45
Enter your choice:
5
```

(3) Test Case 3:

```
Select one of the following options:
1: readin()
2: search()
3: addEmployee()
4: print()
5: exit()
Enter your choice:
1
Enter name:
Hui Siu Cheung
Enter tel:
12345678
Enter id:
11
Enter salary:
123.45
Enter name:
Kenny B
Enter tel:
23456789
Enter id:
12
Enter salary:
1234.45
Enter name:
```

```
#
Enter your choice:
4
The current employee list:
Hui Siu Cheung 12345678 11 123.45
Kenny B 23456789 12 1234.45
Enter your choice:
2
Enter search name:
Kenny BB
Name not found!
Enter your choice:
5
```

(4) Test Case 4:

```
Select one of the following options:
1: readin()
2: search()
3: addEmployee()
4: print()
5: exit()
Enter your choice:
1
Enter name:
Hui Siu Cheung
Enter tel:
12345678
Enter id:
11
Enter salary:
123.45
Enter name:
Kenny B
Enter tel:
23456789
Enter id:
12
Enter salary:
1234.45
Enter name:
#
Enter your choice:
4
The current employee list:
Hui Siu Cheung 12345678 11 123.45
Kenny B 23456789 12 1234.45
Enter your choice:
3
Enter new name:
Kenny Tan
Enter tel:
12344321
Enter id:
13
Enter salary:
2345.67
Added at position: 2
Enter your choice:
4
The current employee list:
Hui Siu Cheung 12345678 11 123.45
```

```
Kenny B 23456789 12 1234.45
Kenny Tan 12344321 13 2345.67
Enter your choice:
5
```