WEEK 6 LAB - STRUCTURES

1. **(computeExp)** A structure is defined to represent an arithmetic expression:

```
typedef struct {
      float operand1, operand2;
      char op;        /* operator '+','-','*' or '/'  */
} bexpression;
```

(a) Write a C function that computes the value of an expression and returns the result. For example, the function will return the value of 4/2 if in the structure passed to it, operand1 is 4, operator is '/' and operand2 is 2. The function prototype is given as:

```
float compute1(bexpression expr);
```

(b) Write another C function that performs the same computation with the following function prototype:

```
float compute2(bexpression *expr);
```

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

```
#include <stdio.h>
typedef struct {
   float operand1, operand2;
   char op;
} bexpression;
float compute1(bexpression expr);
float compute2(bexpression *expr);
int main()
{
   bexpression e;
   int choice;

   printf("Select one of the following options: \n");
   printf("1: compute1()\n");
   printf("2: compute2()\n");
   printf("3: exit()\n");
   do {
      printf("Enter your choice: \n");
      scanf("%d", &choice);
      switch (choice) {
         case 1:
            printf("Enter expression (op1 op2 op): \n");
            scanf("%f %f %c", &e.operand1, &e.operand2, &e.op);
            printf("compute1(): %.2f\n", compute1(e));
            break;
         case 2:
            printf("Enter expression (op1 op2 op): \n");
            scanf("%f %f %c", &e.operand1, &e.operand2, &e.op);
            printf("compute2(): %.2f\n", compute2(&e));
            break;
      }
   } while (choice < 3);
   return 0;
}
float compute1(bexpression expr)
```

```
{
   /* Write your program code here */
}
float compute2(bexpression *expr)
{
   /* Write your program code here */
}
```

Some sample input and output sessions are given below:

(1) Test Case 1:
```
Select one of the following options:
1: compute1()()
2: compute2()
3: exit()
Enter your choice:
1
Enter expression (op1 op2 op) :
5 8 +
compute1(): 13.00
```

(2) Test Case 2:
```
Select one of the following options:
1: compute1()
2: compute2()
3: exit()
Enter your choice:
2
Enter expression (op1 op2 op) :
8 5 /
compute2(): 1.60
```

(3) Test Case 3:
```
Select one of the following options:
1: compute1()
2: compute2()
3: exit()
Enter your choice:
1
Enter expression (op1 op2 op) :
5 8 *
compute1(): 40.00
```

(4) Test Case 4:
```
Select one of the following options:
1: compute1()
2: compute2()
3: exit()
Enter your choice:
2
Enter expression (op1 op2 op) :
8 5 -
compute2(): 3.00
```

2. (**phoneBook**) Write a C program that implements the following three functions:

   a. 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.

b. The function `printPB()` prints the phonebook information on the display. It will print the message “`Empty phonebook`“ if the phonebook list is empty.

c. The function `search()` finds the telephone number of an input name *`target`*, and then prints the name and telephone number on the screen. If the input name cannot be found, then it will print an appropriate error message. The prototypes of the two functions are given below:

The prototypes of the three functions are given below:

```
void printPB(PhoneBk *pb, int size);
int readin(PhoneBk *pb);
void search(PhoneBk *pb, int size, char *target);
```

The structure definition for `PhoneBk` is given below:

```
typedef struct {
   char name[20];
   char telno[20];
} PhoneBk;
```

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

```
#include <stdio.h>
#include <string.h>
#define MAX 100
typedef struct {
   char name[20];
   char telno[20];
} PhoneBk;
void printPB(PhoneBk *pb, int size);
int readin(PhoneBk *pb);
void search(PhoneBk *pb, int size, char *target);
int main()
{
   PhoneBk s[MAX];
   char t[20], *p;
   int size=0, choice, dummychar;

   printf("Select one of the following options: \n");
   printf("1: readin()\n");
   printf("2: search()\n");
   printf("3: printPB()\n");
   printf("4: exit()\n");
   do {
      printf("Enter your choice: \n");
      scanf("%d", &choice);
      switch (choice) {
         case 1:
            scanf("%c", &dummychar);
            size = readin(s);
            break;
         case 2:
            scanf("%c", &dummychar);
            printf("Enter search name: \n");
```

```
               fgets(t, 20, stdin);
               if (p=strchr(t,'\n')) *p = '\0';
               search(s,size,t);
               break;
            case 3:
               printPB(s, size);
               break;
         }
      } while (choice < 4);
      return 0;
   }
   void printPB(PhoneBk *pb, int size)
   {
      /* Write your code here */
   }
   int readin(PhoneBk *pb)
   {
      /* Write your code here */
   }
   void search(PhoneBk *pb, int size, char *target)
   {
      /* Write your code here */
   }
```

Some test input and output sessions are given below:

(1) Test Case 1:

```
Select one of the following options:
1: readin()
2: search()
3: printPB()
4: exit()
Enter your choice:
1
Enter name:
Hui Siu Cheung
Enter tel:
1234567
Enter name:
Philip Fu
Enter tel:
2345678
Enter name:
Chen Jing
Enter tel:
3456789
Enter name:
#
Enter your choice:
3
The phonebook list:
Name: Hui Siu Cheung
Telno: 1234567
Name: Philip Fu
Telno: 2345678
Name: Chen Jing
```

```
Telno: 3456789
Enter your choice:
4
```

(2) Test Case 2:

```
Enter your choice:
2
Enter search name:
Philip Fu
Name = Philip Fu, Tel = 2345678
Enter your choice:
4
```

(3) Test Case 3:

```
Enter your choice:
2
Enter search name:
Tommy Fu
Name not found!
Enter your choice:
4
```

(4) Test Case 4:

```
Select one of the following options:
1: readin()
2: search()
3: printPB()
4: exit()
Enter your choice:
1
Enter name:
#
Enter your choice:
3
The phonebook list:
Empty phonebook
Enter your choice:
4
```