Lab 5 - Structures

Note: You are required to submit your lab code as part of assignment submission for grading via APAS.

1. 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 follows:

```
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 code here */
}
float compute2(bexpression *expr)
{
    /* Write your 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
Enter your choice:
2
Enter expression (op1 op2 op):
5 8 +
compute2(): 13.00
Enter your choice:
3
```

(2) Test Case 2:

```
Select one of the following options:
1: compute1()
2: compute2()
3: exit()
Enter your choice:
```

1
Enter expression (op1 op2 op):
8 5 /
compute1(): 1.60
Enter your choice:
2
Enter expression (op1 op2 op):
8 5 /
compute2(): 1.60
Enter your choice:
3

(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
Enter your choice:
2
Enter expression (op1 op2 op):
5 8 *
compute2(): 40.00
Enter your choice:
3

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

2. Given the following structure definition, write the code for the functions getInput(), mayTakeLeave() and printList() with the following function prototypes:

Given the following structure definition, write the code for the functions getInput(), mayTakeLeave() and printList() with the following function prototypes:

```
typedef struct {
  int id;              /* staff identifier */
  int totalLeave;   /* the total number of days of leave allowed */
  int leaveTaken;   /* the number of days of leave taken so far */
} leaveRecord;
```

(a) void getInput(leaveRecord list[ ], int *n);

Each line of the input has three integers representing one staff identifier, his/her total number of days of leave allowed and his/her number of days of leave taken so far respectively. The function will read the data into the array *list* until end of input and returns the number of records read through *n*.

(b) int mayTakeLeave(leaveRecord list[ ], int id, int leave, int n);

It returns 1 if a leave application for *leave* days is approved. Staff member with identifier *id* is applying for *leave* days of leave. *n* is the number of staff in *list*. Approval will be given if the leave taken so far plus the number of days applied for is less than or equal to his total number of *leave* days allowed. If approval is not given, it returns 0. It will return -1 if no one in *list* has identifier *id*.

(c) void printList(leaveRecord list[ ], int n);

It prints the *list* of leave records of each staff. *n* is the number of staff in *list*.

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

```
#include <stdio.h>
#define INIT_VALUE 1000
typedef struct {
  int id;              /* staff identifier */
  int totalLeave;   /* the total number of days of leave allowed */
  int leaveTaken;   /* the number of days of leave taken so far */
} leaveRecord;
int mayTakeLeave(leaveRecord list[], int id, int leave, int n);
void getInput(leaveRecord list[], int *n);
void printList(leaveRecord list[], int n);
int main()
{
  leaveRecord listRec[10];
  int len;
  int id, leave, canTake=INIT_VALUE;
  int choice;
```

```
  printf("Select one of the following options: \n");
  printf("1: getInput()\n");
  printf("2: printList()\n");
  printf("3: mayTakeLeave()\n");
  printf("4: exit()\n");
  do {
    printf("Enter your choice: \n");
    scanf("%d", &choice);
    switch (choice) {
      case 1:
        getInput(listRec, &len);
        printList(listRec, len);
        break;
      case 2:
        printList(listRec, len);
        break;
      case 3:
        printf("Please input id, leave to be taken: \n");
        scanf("%d %d", &id, &leave);
        canTake = mayTakeLeave(listRec, id, leave, len);
        if (canTake == 1)
          printf("The staff %d can take leave\n", id);
        else if (canTake == 0)
          printf("The staff %d cannot take leave\n", id);
        else if (canTake == -1)
          printf("The staff %d is not in the list\n", id);
        else
          printf("Error!");
        break;
    }
  } while (choice < 4);
  return 0;
}
void printList(leaveRecord list[], int n)
{
  int p;

  printf("The staff list:\n");
  for (p = 0; p < n; p++)
    printf ("id = %d, totalleave = %d, leave taken = %d\n",
      list[p].id, list[p].totalLeave, list[p].leaveTaken);
}
void getInput(leaveRecord list[], int *n)
{
     /* Write your program code here */
}
int mayTakeLeave(leaveRecord list[], int id, int leave, int n)
```

```
{
    /* 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: getInput()
2: printList()
3: mayTakeLeave()
4: exit()
Enter your choice:
1
Enter the number of staff records:
2
Enter id, totalleave, leavetaken:
11 28 25
Enter id, totalleave, leavetaken:
12 28 6
The staff list:
id = 11, totalleave = 28, leave taken = 25
id = 12, totalleave = 28, leave taken = 6
Enter your choice:
3
Please input id, leave to be taken:
11 6
The staff 11 cannot take leave
Enter your choice:
4
```

(2) Test Case 2:

```
Select one of the following options:
1: getInput()
2: printList()
3: mayTakeLeave()
4: exit()
Enter your choice:
1
Enter the number of staff records:
2
Enter id, totalleave, leavetaken:
11 28 25
Enter id, totalleave, leavetaken:
12 28 6
The staff list:
id = 11, totalleave = 28, leave taken = 25
id = 12, totalleave = 28, leave taken = 6
```

Enter your choice:
3
Please input id, leave to be taken:
12 6
The staff 12 can take leave
Enter your choice:
4

(3) Test Case 3:

Select one of the following options:
1: getInput()
2: printList()
3: mayTakeLeave()
4: exit()
Enter your choice:
1
Enter the number of staff records:
2
Enter id, totalleave, leavetaken:
11 28 25
Enter id, totalleave, leavetaken:
12 28 6
The staff list:
id = 11, totalleave = 28, leave taken = 25
id = 12, totalleave = 28, leave taken = 6
Enter your choice:
3
Please input id, leave to be taken:
13 6
The staff 13 is not in the list
Enter your choice:
4

(4) Test Case 4:

Select one of the following options:
1: getInput()
2: printList()
3: mayTakeLeave()
4: exit()
Enter your choice:
1
Enter the number of staff records:
2
Enter id, totalleave, leavetaken:
11 28 25
Enter id, totalleave, leavetaken:
12 28 6
The staff list:

id = 11, totalleave = 28, leave taken = 25
id = 12, totalleave = 28, leave taken = 6
Enter your choice:
4