Tutorial 11

ZHANG Xinyun

- Quiz 1 exercises

Exercise 1

Michael has written the following header files stack.h and queue.h for the ADTs of stack of integers and queue of integers. He has correctly implemented the ADTs in stack.c and queue.c.

```
/* File: stack.h */
/* Author: Michael */

typedef struct stackCDT *stackADT;
typedef int stackElementT;

stackADT EmptyStack(void);
void Push(stackADT, stackElementT);
stackElementT Pop(stackADT);
int StackIsEmpty(stackADT);
```

```
/* File: queue.h */
/* Author: Michael */

typedef struct queueCDT *queueADT;
typedef int queueElementT;

queueADT EmptyQueue(void);
void Enqueue(queueADT, queueElementT);
queueElementT Dequeue(queueADT);
int QueueIsEmpty(queueADT);
```

Exercise 1

a) **(20 marks)** What will be printed if the following program is run?

```
#include <stdio.h>
#include <stdlib.h>
#include "stack.h"
#include "queue.h"

main() {
  stackADT S1 = EmptyStack();
  queueADT Q1 = EmptyQueue();
  Push(S1, 4); Push(S1, 5); Push(S1, 6);
  Enqueue(Q1, Pop(S1)); Enqueue(Q1, 7);
  Push(S1, 1 + Dequeue(Q1));
  while (!StackIsEmpty(S1)) printf("%d  ", Pop(S1)); printf("\n");
  while (!QueueIsEmpty(Q1)) printf("%d  ", Dequeue(Q1)); printf("\n");
}
```

Exercise 1

a) **(20 marks)** What will be printed if the following program is run?

s1

q1

```
#include <stdio.h>
#include <stdlib.h>
#include "stack.h"
#include "queue.h"

main() {
  stackADT S1 = EmptyStack();
  queueADT Q1 = EmptyQueue();
  Push(S1, 4); Push(S1, 5); Push(S1, 6);
  Enqueue(Q1, Pop(S1)); Enqueue(Q1, 7);
  Push(S1, 1 + Dequeue(Q1));
  while (!StackIsEmpty(S1)) printf("%d  ", Pop(S1)); printf("\n");
  while (!QueueIsEmpty(Q1)) printf("%d  ", Dequeue(Q1)); printf("\n");
}
```

Exercise 1

a) **(20 marks)** What will be printed if the following program is run?

s1

q1

```
#include <stdio.h>
#include <stdlib.h>
#include "stack.h"
#include "queue.h"

main() {
  stackADT S1 = EmptyStack();
  queueADT Q1 = EmptyQueue();
  Push(S1, 4); Push(S1, 5); Push(S1, 6);
  Enqueue(Q1, Pop(S1)); Enqueue(Q1, 7);
  Push(S1, 1 + Dequeue(Q1));
  while (!StackIsEmpty(S1)) printf("%d  ", Pop(S1)); printf("\n");
  while (!QueueIsEmpty(Q1)) printf("%d  ", Dequeue(Q1)); printf("\n");
}
```

Exercise 1

a) **(20 marks)** What will be printed if the following program is run?

5 4	s1
6	q1

```
#include <stdio.h>
#include <stdlib.h>
#include "stack.h"
#include "queue.h"

main() {
  stackADT S1 = EmptyStack();
  queueADT Q1 = EmptyQueue();
  Push(S1, 4); Push(S1, 5); Push(S1, 6);
  Enqueue(Q1, Pop(S1)); Enqueue(Q1, 7);
  Push(S1, 1 + Dequeue(Q1));
  while (!StackIsEmpty(S1)) printf("%d  ", Pop(S1)); printf("\n");
  while (!QueueIsEmpty(Q1)) printf("%d  ", Dequeue(Q1)); printf("\n");
}
```

Exercise 1

a) **(20 marks)** What will be printed if the following program is run?

5 4	s1
6 7	q1

```
#include <stdio.h>
#include <stdlib.h>
#include "stack.h"
#include "queue.h"

main() {
  stackADT S1 = EmptyStack();
  queueADT Q1 = EmptyQueue();
  Push(S1, 4); Push(S1, 5); Push(S1, 6);
  Enqueue(Q1, Pop(S1)); Enqueue(Q1, 7);
  Push(S1, 1 + Dequeue(Q1));
  while (!StackIsEmpty(S1)) printf("%d  ", Pop(S1)); printf("\n");
  while (!QueueIsEmpty(Q1)) printf("%d  ", Dequeue(Q1)); printf("\n");
}
```

Exercise 1

a) **(20 marks)** What will be printed if the following program is run?

s1

q1

```
#include <stdio.h>
#include <stdlib.h>
#include "stack.h"
#include "queue.h"

main() {
  stackADT S1 = EmptyStack();
  queueADT Q1 = EmptyQueue();
  Push(S1, 4); Push(S1, 5); Push(S1, 6);
  Enqueue(Q1, Pop(S1)); Enqueue(Q1, 7);
  Push(S1, 1 + Dequeue(Q1));
  while (!StackIsEmpty(S1)) printf("%d  ", Pop(S1)); printf("\n");
  while (!QueueIsEmpty(Q1)) printf("%d  ", Dequeue(Q1)); printf("\n");
}
```

Exercise 1

a) **(20 marks)** What will be printed if the following program is run?

7 5 4 s1

7 q1

```
#include <stdio.h>
#include <stdlib.h>
#include "stack.h"
#include "queue.h"

main() {
  stackADT S1 = EmptyStack();
  queueADT Q1 = EmptyQueue();
  Push(S1, 4); Push(S1, 5); Push(S1, 6);
  Enqueue(Q1, Pop(S1)); Enqueue(Q1, 7);
  Push(S1, 1 + Dequeue(Q1));
  while (!StackIsEmpty(S1)) printf("%d  ", Pop(S1)); printf("\n");
  while (!QueueIsEmpty(Q1)) printf("%d  ", Dequeue(Q1)); printf("\n");
}
```

Exercise 1

c) **(15 marks)** Complete the stackDepth function. Note that the argument should not be altered. (Hint: count the number of elements you can pop from the stack.)

```
int StackDepth(stackADT S){
    int depth = 0;
    stackADT helperStack = EmptyStack();
    while(!StackIsEmpty(S)){
        Push(helperStack, Pop(S));
        depth++;
    }
    while(!StackIsEmpty(helperStack))
        Push(S, Pop(helperStack));
    return depth;
}
```

Exercise 2

Teresa defines the symbol table ADT in the following header file symtab.h. The keys are character strings (char*) and the values are pointed to by void pointers (void*).

```
/*
 * File: symtab.h
 */

typedef struct symtabCDT *symtabADT;

symtabADT EmptySymbolTable(void);
void Enter(symtabADT, char*, void*);
void *Lookup(symtabADT, char*);
int SymTabIsEmpty(symtabADT);
```

Exercise 2

Teresa correctly implements the symbol table ADT using Hash tables in file symtab.c. She uses open addressing hashing as the collision resolution scheme. Part of the implementation file symtab.c is shown below.

```
/*
 * File: symtab.c
 * By: Teresa
 */

#include <stdlib.h>
#include <string.h>

typedef struct cellT { char *key; void *value; struct cellT *next; } cellT;

struct symtabCDT { cellT *bucket[23]; }; /* Note the number of buckets */

int Hash(char *s) { … } // Not shown.

symtabADT EmptySymbolTable(void) {
   symtabADT table = (symtabADT)malloc(sizeof(*table));
   for (int i=0; i<23; i++) table->bucket[i] = NULL;
   return table;
}

void Enter(symtabADT table, char *key, void *value) { … } // Not shown.

void *Lookup(symtabADT table, char *key) { … } // Not shown.
```

Exercise 2

a) **(20 marks)** The function NrOfEntriesEntered accepts one argument of the type symtabADT. It returns the number of entries already entered in the symbol table as an int value. Write the function NrOfEntriesEntered for Teresa's implementation.

```
int NrOfEntriesEntered(symtabADT S) { … … }
```

Exercise 2

a) **(20 marks)** The function NrOfEntriesEntered accepts one argument of the type symtabADT. It returns the number of entries already entered in the symbol table as an int value. Write the function NrOfEntriesEntered for Teresa's implementation.

int NrOfEntriesEntered(symtabADT S) { … … }

```
int NrOfEntriesEntered(Symtab S){
    int cnt = 0;
    for(int i = 0; i < 23; i++){
        if(S->buckets[i] != NULL)
            cnt++;
    }
    return cnt;
}
```

Exercise 2

Teresa uses the following function as the hash function in her implementation file symtab.c.

```
/* Author: Teresa */

int Hash(char *s) {
   unsigned long hashcode = 0;
   for (int i=0; s[i]!='\0'; i++) hashcode = (hashcode<<7) + missing part 1 ;
   return missing part 2 ;
}
```

b) **(5 marks)** What should Teresa write in the place marked ***missing part 1*** in the above hash function? s[i]

c) **(5 marks)** What should Teresa write in the place marked ***missing part 2*** in the above hash function?

Exercise 2

Teresa uses the following function as the hash function in her implementation file symtab.c.

```
/* Author: Teresa */

int Hash(char *s) {
   unsigned long hashcode = 0;
   for (int i=0; s[i]!='\0'; i++) hashcode = (hashcode<<7) + missing part 1 ;
   return missing part 2 ;
}
```

b) **(5 marks)** What should Teresa write in the place marked ***missing part 1*** in the above hash function?

s[i]

c) **(5 marks)** What should Teresa write in the place marked ***missing part 2*** in the above hash function?

(int) hashcode % 23

Exercise 2

Entries	Key	Primary Hash code	Secondary Hash code
First entry entered	"Shui Chuen O"	18	8
Second entry entered	"Shek Mun"	18	1
Third entry entered	"Fung Wo"	17	2
Fourth entry entered	"Kwong Yuen"	15	4
Fifth entry entered	"Pok Hong"	20	10
Sixth entry entered	"Sha Kok"	20	10

d) **(2 marks)** In which bucket will the entry with key "Shui Chuen O" be stored?
e) **(2 marks)** In which bucket will the entry with key "Shek Mun" be stored?
f) **(2 marks)** In which bucket will the entry with key "Fung Wo" be stored?
g) **(2 marks)** In which bucket will the entry with key "Kwong Yuen" be stored?
h) **(2 marks)** In which bucket will the entry with key "Pok Hong" be stored?
i) **(2 marks)** In which bucket will the entry with key "Sha Kok" be stored?

Exercise 2

Entries	Key	Primary Hash code	Secondary Hash code
First entry entered	"Shui Chuen O"	18	8
Second entry entered	"Shek Mun"	18	1
Third entry entered	"Fung Wo"	17	2
Fourth entry entered	"Kwong Yuen"	15	4
Fifth entry entered	"Pok Hong"	20	10
Sixth entry entered	"Sha Kok"	20	10

d: (18 + 0 * 8) % 23=18

d) **(2 marks)** In which bucket will the entry with key "Shui Chuen O" be stored?
e) **(2 marks)** In which bucket will the entry with key "Shek Mun" be stored?
f) **(2 marks)** In which bucket will the entry with key "Fung Wo" be stored?
g) **(2 marks)** In which bucket will the entry with key "Kwong Yuen" be stored?
h) **(2 marks)** In which bucket will the entry with key "Pok Hong" be stored?
i) **(2 marks)** In which bucket will the entry with key "Sha Kok" be stored?

Exercise 2

Entries	Key	Primary Hash code	Secondary Hash code
First entry entered	"Shui Chuen O"	18	8
Second entry entered	"Shek Mun"	18	1
Third entry entered	"Fung Wo"	17	2
Fourth entry entered	"Kwong Yuen"	15	4
Fifth entry entered	"Pok Hong"	20	10
Sixth entry entered	"Sha Kok"	20	10

e:
(18 + 0 * 8) % 23=18->occupied
(18 + 1 * 1) % 23=19

d) **(2 marks)** In which bucket will the entry with key "Shui Chuen O" be stored?
e) **(2 marks)** In which bucket will the entry with key "Shek Mun" be stored?
f) **(2 marks)** In which bucket will the entry with key "Fung Wo" be stored?
g) **(2 marks)** In which bucket will the entry with key "Kwong Yuen" be stored?
h) **(2 marks)** In which bucket will the entry with key "Pok Hong" be stored?
i) **(2 marks)** In which bucket will the entry with key "Sha Kok" be stored?

Exercise 2

Entries	Key	Primary Hash code	Secondary Hash code
First entry entered	"Shui Chuen O"	18	8
Second entry entered	"Shek Mun"	18	1
Third entry entered	"Fung Wo"	17	2
Fourth entry entered	"Kwong Yuen"	15	4
Fifth entry entered	"Pok Hong"	20	10
Sixth entry entered	"Sha Kok"	20	10

d) **(2 marks)** In which bucket will the entry with key "Shui Chuen O" be stored?
e) **(2 marks)** In which bucket will the entry with key "Shek Mun" be stored?
f) **(2 marks)** In which bucket will the entry with key "Fung Wo" be stored?
g) **(2 marks)** In which bucket will the entry with key "Kwong Yuen" be stored?
h) **(2 marks)** In which bucket will the entry with key "Pok Hong" be stored?
i) **(2 marks)** In which bucket will the entry with key "Sha Kok" be stored?

f:
(17 + 0 * 2) % 23=17

Exercise 2

Entries	Key	Primary Hash code	Secondary Hash code
First entry entered	"Shui Chuen O"	18	8
Second entry entered	"Shek Mun"	18	1
Third entry entered	"Fung Wo"	17	2
Fourth entry entered	"Kwong Yuen"	15	4
Fifth entry entered	"Pok Hong"	20	10
Sixth entry entered	"Sha Kok"	20	10

d) **(2 marks)** In which bucket will the entry with key "Shui Chuen O" be stored?
e) **(2 marks)** In which bucket will the entry with key "Shek Mun" be stored?
f) **(2 marks)** In which bucket will the entry with key "Fung Wo" be stored?
g) **(2 marks)** In which bucket will the entry with key "Kwong Yuen" be stored?
h) **(2 marks)** In which bucket will the entry with key "Pok Hong" be stored?
i) **(2 marks)** In which bucket will the entry with key "Sha Kok" be stored?

g:
(15 + 0 * 4) % 23=15

Exercise 2

Entries	Key	Primary Hash code	Secondary Hash code
First entry entered	"Shui Chuen O"	18	8
Second entry entered	"Shek Mun"	18	1
Third entry entered	"Fung Wo"	17	2
Fourth entry entered	"Kwong Yuen"	15	4
Fifth entry entered	"Pok Hong"	20	10
Sixth entry entered	"Sha Kok"	20	10

h:
(20 + 0 * 10) % 23=20

d) **(2 marks)** In which bucket will the entry with key "Shui Chuen O" be stored?
e) **(2 marks)** In which bucket will the entry with key "Shek Mun" be stored?
f) **(2 marks)** In which bucket will the entry with key "Fung Wo" be stored?
g) **(2 marks)** In which bucket will the entry with key "Kwong Yuen" be stored?
h) **(2 marks)** In which bucket will the entry with key "Pok Hong" be stored?
i) **(2 marks)** In which bucket will the entry with key "Sha Kok" be stored?

Exercise 2

Entries	Key	Primary Hash code	Secondary Hash code
First entry entered	"Shui Chuen O"	18	8
Second entry entered	"Shek Mun"	18	1
Third entry entered	"Fung Wo"	17	2
Fourth entry entered	"Kwong Yuen"	15	4
Fifth entry entered	"Pok Hong"	20	10
Sixth entry entered	"Sha Kok"	20	10

d) **(2 marks)** In which bucket will the entry with key "Shui Chuen O" be stored?
e) **(2 marks)** In which bucket will the entry with key "Shek Mun" be stored?
f) **(2 marks)** In which bucket will the entry with key "Fung Wo" be stored?
g) **(2 marks)** In which bucket will the entry with key "Kwong Yuen" be stored?
h) **(2 marks)** In which bucket will the entry with key "Pok Hong" be stored?
i) **(2 marks)** In which bucket will the entry with key "Sha Kok" be stored?

i:
(20 + 0 * 10) % 23=20->occupied
(20 + 1 * 10) % 23=7

Exercise 2

In order to better understand Hash tables, Katherine decides to implement symbol table ADT again using separate chaining as the collision resolution scheme. She still uses 23 buckets in her implementation.

Assume that Katherine writes a testing program and enters the same number of entries to an empty symbol table in the same order as before. She does not change the Hash functions used in her double hashing implementation, therefore, the keys of the entries and their respective primary and secondary hash codes are same as before, as shown in the following table (hint: not all information in the following table is useful).

Entries	Key	Primary Hash code	Secondary Hash code
First entry entered	"Shui Chuen O"	18	8
Second entry entered	"Shek Mun"	18	1
Third entry entered	"Fung Wo"	17	2
Fourth entry entered	"Kwong Yuen"	15	4
Fifth entry entered	"Pok Hong"	20	10
Sixth entry entered	"Sha Kok"	20	10

j) **(8 marks)** Draw a diagram of Katherine's new implementation after all these entries are entered, to show where the entries with different keys are stored. Remember to show the bucket numbers (bucket 0, bucket 1, … etc.) next to each bucket.

Exercise 2

Entries	Key	Primary Hash code	Secondary Hash code
First entry entered	"Shui Chuen O"	18	8
Second entry entered	"Shek Mun"	18	1
Third entry entered	"Fung Wo"	17	2
Fourth entry entered	"Kwong Yuen"	15	4
Fifth entry entered	"Pok Hong"	20	10
Sixth entry entered	"Sha Kok"	20	10