NANYANG TECHNOLOGICAL UNIVERSITY

SEMESTER 2 EXAMINATION 2014-2015

CE1007/CZ1007 – DATA STRUCTURES

Apr/May 2015 Time Allowed: 2 hours

INSTRUCTIONS

1. This paper contains 4 questions and comprises 9 pages.
2. Answer **ALL** questions.
3. This is a closed-book examination.
4. All questions carry equal marks.

---

1. (a) What are the outputs of the following code?

(i)
```
#include <stdio.h>
int main() {
   int a[4][4]={  {1,2,-3,-4},
                  {5,-4,-3,2},
                  {2,0,-3,2},
                  {-6,5,-4,3}  };
   int i,j,s=0;
   for (i=0; i<4; i++) {
      for (j=0; j<4; j++) {
         if (a[i][j]<0) continue;
         if (a[i][j]==0) break;
         s+=a[i][j];
      }
      printf("%d\n",s);
   }
   return 0;
}
```

(4 marks)

Note: Question No. 1 continues on Page 2

(ii)
```
#include <stdio.h>
void f1(int *a, int *b);
int main() {
   int a=1,b=2,c=3;
   f1(&a,&b); printf("%d,%d\n",a,b);
   f1(&b,&c); printf("%d,%d\n",b,c);
   f1(&c,&a); printf("%d,%d\n",c,a);
   return 0;
}
void f1(int *a, int *b) {
   *a=*a+*b;
   *b=*a-*b;
   *a=*a-*b;
}
```

(4 marks)

(b) Find four errors in each of the following code. Correct the errors by stating the line numbers and rewriting the corrected statements in your answer book.

(i)
```
/* This program reads in a sequence of
   positive numbers, terminated by 0, and
   computes and prints the average of the
   positive numbers (excluding 0). */
 1 #include <stdio.h>
 2 int main() {
 3    int s[100];
 4    int i=0,c=0;
 5    int sum=0;
 6    do {
 7       scanf("%d",&s[i]);
 8    } while (s[i++] != 0);
 9    i=0;
10    while (s[i] = 0) {
11       printf("%d ",s[i]);
12       if (s[i]>0) {
13          sum += s[i];
14          c++;
15       }
16       i++;
17    }
18    sum \= c;
19    printf("%f\n",c);
20    return 0;
21 }
```

(4 marks)

Note: Question No. 1 continues on Page 3

(ii)

```
/* This program finds the smallest string
   from an array of 5 strings according to
   the ascending alphabetical order. */
#include <stdio.h>
#include <string.h>
int main() {
   char *name[5] = {"Windows", "Word",
      "Excel", "Foxpro", "Visual Basic"};
   char temp;
   int i;
   temp=name[0];
   for (i=1; i<=5; i++)
      if (temp>name[i])
         temp=name[i];
   printf("%s\n",*temp);
   return 0;
}
```

(4 marks)

(c) The following program reads in two character strings `a` and `b`, concatenates them to form a new string, and prints the resultant string to the screen. For example, if `a` = "Nanyang " and `b` = "Technological University", then the new string is "Nanyang Technological University". Write the missing code segment in the program. Note that you should not use any standard C string functions and declare any other variables in the program.

```
#include <stdio.h>
int main() {
   char a[80],b[80],*p,*q;

   gets(a);
   gets(b);
   /* missing code segment here */
   return 0;
}
```

(9 marks)

2. (a) What are the outputs of the following code?

(i)
```
#include <stdio.h>
int f2(int x, int *y);
int main() {
   int x=3, y=5;
   printf("%d\n", f2(x,&y));
   return 0;
}
int f2(int x, int *y)
{
   if (x<=0 || *y<=0)
      return 1;
   else {
      *y -=2;
      return 5*f2(x-1,y);
   }
}
```

(3 marks)

(ii)
```
#include <stdio.h>
long f3(int n);
int main(){
   printf("f3(): %d\n", f3(5));
   return 0;
}
long f3(int n){
   switch (n) {
      case 0:
         return 0;
      case 1: case 2:
         return 1;
   }
   return (f3(n-1)+f3(n-2));
}
```

(4 marks)

(b) Find four errors in each of the following code. Correct the errors by stating the line numbers and rewriting the corrected statements in your answer book.

Note: Question No. 2 continues on Page 5

(i)

```
/* f4() counts the number of upper and
   lower case letters from a string. */
#include <stdio.h>
void f4(char *s, int a, int b);
int main() {
   char s[80]; int upper=0,lower=0;
   gets(s); f4(s,&upper,&lower);
   printf("%d %d\n",upper,lower);
   return 0;
}
void f4(char *s, int a, int b) {
   while (*s) {
      if (*s>='A' && *s<='Z') a++;
      if (*s>='a' && *s<='z') b++;
      s++;
   }
}
```

(4 marks)

(ii)

```
/* f5() finds the largest character from
   a string, and moves it to the beginning
   of the string. E.g., if str is "aedcb"
   before calling f5(), then str will be
   "eadcb" after executing f5(). */
#include <stdio.h>
void f5(char *p);
int main() {
   char *str;
   gets(str); f5(str); puts(str);
   return 0;
}
void f5(char *p) {
   char max,*q; int i=0;
   while (p[i] != '\0') {
      if (max<p[i]) {
         max=p[i];
         p=q+i;
      }
      i++;
   }
   while (q<p) {
      q=*(q-1);
      q--;
   }
   p[0]=max;
}
```

(6 marks)

Note: Question No. 2 continues on Page 6

(c) In the following program, the **recursive** function `rLookup()` returns the subscript of the first appearance of a target number in the array. The function takes in three arguments `ar[]`, `n` and `target`, which are an array of integers, an integer specifying the size of the array and the target number respectively. For example, if the array contains, in order, 1, 2, 3, 4, 5, and the target number is 2, then `rLookup()` will return 1. If the target number is not in the array, then `rLookup()` will return –1. Write the missing code segment in the program. Note that you should not declare any other variables in the program.

```
#include <stdio.h>
int rLookup(int ar[], int n, int target);
int main(){
   int a[80];
   int target, i, size;
   printf("Enter array size: ");
   scanf("%d", &size);
   printf("Enter %d numbers: ", size);
   for (i=0; i < size; i++)
      scanf("%d", &a[i]);
   printf("Enter the target number: ");
   scanf("%d", &target);
   printf("%d", rLookup(a, size, target));
   return 0;
}
int rLookup(int ar[], int n, int target){
   int i;
   if (n == 1) {
      if (ar[0] == target)
         return 0;
      else
         return -1;
   }
   else {
      /* missing code segment here */
   }
}
```

(8 marks)

3. (a) The following program asks the user how many integers will be entered, and then reads in the integers from the user.

(i) Fill the missing codes of Line 8 and Line 13.

(4 marks)

(ii) If Line 11 is replaced by "`for (i=0; i<=n; i++)`", what problem will it cause?

(2 marks)

```
1   #include <stdlib.h>
2   #include <stdio.h>
3   int main(){
4      int n, i;
5      int *intArr;
6      printf("How many integers do you have? ");
7      scanf("%d", &n);
8      intArr = malloc(_____________________________);
9      if (intArr == NULL) printf("error.\n");
10    /*Loop over array & store integers entered*/
11     for (i=0; i<n; i++)
12         scanf("%d",&intArr[i]);
13     _________________;/*free the memory space of
                          intArr*/
14     return 0;
15  }
```

(b) Consider an empty stack S of integers. Let integers be pushed into the stack S in the order of **1→2→3**. Assume U and O indicate the push() and pop() operations respectively. For example, the operation sequence UOUUOO for the stack S outputs (pops) integers in the order: 1, 3, 2.

(i) What is the output if the operation sequence is UUOO?

(2 marks)

(ii) List all possible *6-operation sequences* and the corresponding outputs. A *6-operation sequence* means a sequence with 6 operations, e.g., UUUOOO.

(5 marks)

(iii) State the main difference between a queue and a stack.

(2 marks)

Note: Question No. 3 continues on Page 8

(c) Write the missing code segment of the function `swap2n()` that accepts the head pointer of a linked list of integers. It swaps the **2nd node** with the **last node** of the linked list and **returns the number of nodes** in the linked list. **Swapping of data is not allowed, only pointers should be changed.** Suppose the linked list has at least **4 nodes**. You are **NOT** allowed to call any linked list functions.

```
typedef struct _listnode{
   int item;
  _listnode *next;
} ListNode;

int swap2n(ListNode *head){
   int size=0;          /* size is the number of
                           nodes in the linked list*/
   ListNode *p=head;  /* pointer p is used for
                      traversal of the linked list*/
   ListNode *tmp;     /*for temporary store of some
                      pointers when swapping nodes*/

   /* missing code segment here */

   return size;
}
```

(10 marks)

4. (a) Two trees are shown in Figure Q4(1) and Figure Q4(2).

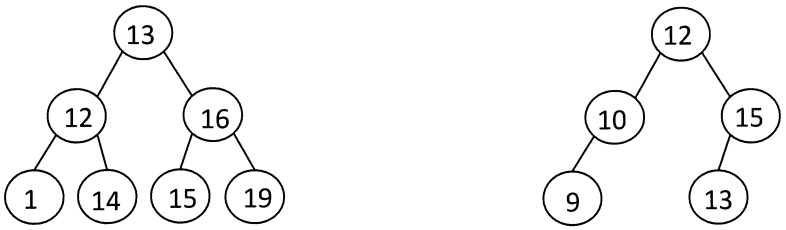
(i) Is the tree in Figure Q4(1) a binary search tree? Justify your answer.

(3 marks)

(ii) Is the tree in Figure Q4(2) a binary search tree? Justify your answer.

(3 marks)

Note: Question No. 4 continues on Page 9



(1) (2)

Figure Q4

(b) A binary tree has 4 nodes that store 'A', 'B', 'C' and 'D' respectively. The in-order traversal visits the tree in the order of ABCD, and the pre-order traversal visits the tree in the order of CABD.

(i) Draw the binary tree structure.

(5 marks)

(ii) Based on the binary tree you draw, what is the node visit order by the post-order traversal?

(3 marks)

(c) Write a **recursive** C function `printParent()` that accepts a pointer `node` to the **root** node and a pointer `x` to a **non-root** node of a binary search tree. It finds the parent node of `x` and prints the parent node's item value. It returns **1** when it is able to find the parent node of `x`, otherwise it returns **0**. The function prototype is given as follows:

```
int printParent(BTNode *node, BTNode *x);
```

A `BTNode` structure is defined as follows:

```
typedef struct _btnode{
   int item;
   struct _btnode *left;
   struct _btnode *right;
} BTNode;
```

(11 marks)

END OF PAPER

CE1007 DATA STRUCTURES
CZ1007 DATA STRUCTURES

Please read the following instructions carefully:

1. **Please do not turn over the question paper until you are told to do so. Disciplinary action may be taken against you if you do so.**

2. You are not allowed to leave the examination hall unless accompanied by an invigilator. You may raise your hand if you need to communicate with the invigilator.

3. Please write your Matriculation Number on the front of the answer book.

4. Please indicate clearly in the answer book (at the appropriate place) if you are continuing the answer to a question elsewhere in the book.