NANYANG TECHNOLOGICAL UNIVERSITY

SEMESTER 2 EXAMINATION 2013-2014

CE1007/CZ1007 – DATA STRUCTURES

Apr/May 2014 Time Allowed: 2 hours

INSTRUCTIONS

1. This paper contains 4 questions and comprises 6 pages.

2. Answer **ALL** questions.

3. This is a closed-book examination.

4. All questions carry equal marks.

---

1. (a) What is the output of the following code?

```
#include <stdio.h>
void fun(int n, int *r) ;
int main()
{
   int m=10,r=1;
   fun(m,&r);
   printf("m=%d,r=%d\n",m,r);
   return 0;
}
void fun(int n, int *r)
{
   *r*=n;
   if (n>0)
      fun(n-4,r);
   printf("n=%d,*r=%d\n",n,*r);
}
```

(7 marks)

Note: Question No. 1 continues on Page 2

(b) In the following program, the function `readin()` reads a number of people's names and their corresponding telephone numbers, passes the data to the calling function via the parameter `p`, and returns the number of names that have been entered. The character '#' is used to indicate the end of user input. 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. Write the two missing code segments `(i)` and `(ii)` in the program. Note that you should not declare any other variables in the program.

```
#include <stdio.h>
#include <string.h>
#define MAX 100
typedef struct {
   char name[10];
   char telno[10];
} PhoneBk;
int readin(PhoneBk *p);
void search(PhoneBk *p, int size, char *target);
int main()
{
   PhoneBk s[MAX];
   char t[10];
   int size;
   size = readin(s);
   printf("Enter search name: ");
   gets(t);
   search(s,size,t);
   return 0;
}
int readin(PhoneBk *p)
{
   int size=0;
   /* (i) missing code segment here */
   return size;
}
void search(PhoneBk *p, int size, char *target)
{
   int i;
   /* (ii) missing code segment here */
}
```

(10 marks)

Note: Question No. 1 continues on Page 3

(c) The following program reads a character string and determines whether or not it is a *palindrome*. A palindrome is a sequence of characters that reads the same forwards and backwards. For example, *abba* and *abcba* are palindromes, but *abcd* is not. Write the missing code segment in the program. Note that you should not declare any other variables in the program.

```
#include <stdio.h>
#include <string.h>
int main()
{
   char str[80],*p1,*p2;
   int n;
   gets(str);
   n=strlen(str);
   /* missing code segment here */
   return 0;
}
```

(8 marks)

2. (a) What is the output of the following code?

```
#include <stdio.h>
int main()
{
   int a[]={2,4,6,8,10},i;
   int *ptr[]={&a[0],&a[1],&a[2],&a[3],&a[4]};
   int *pa,**pb;
   pa=a;
   printf("%d,",*++pa);
   printf("%d\n",*pa++);
   for (i=0;i<5;i++) {
      a[i]=a[i]/2+a[i];
      printf("%d,",a[i]);
   }
   pb=ptr;
   printf("\n%d,", *(*(pb+2)));
   printf("%d", *(*(++pb)));
   return 0;
}
```

(8 marks)

Note: Question No. 2 continues on Page 4

(b) In the following program, the function `max()` takes a one-dimensional array `m` of integers as a parameter. The function returns the largest and second largest numbers of the array to the `main()` function through the two parameters `max1` and `max2` respectively. For example, if the array `a[]={1,7,8,6,9,4,5,2,3}`, then `max1` is 9, and `max2` is 8. Write the missing code segment in the program. Note that you should not declare any other variables in the program.

```
#include <stdio.h>
#define SIZE 9
void max(int m[SIZE], int *max1, int *max2);
int main()
{
   int max1,max2;
   int a[]={1,7,8,6,9,4,5,2,3};
   max(a,&max1,&max2);
   printf("max1=%d,max2=%d\n",max1,max2);
   return 0;
}
void max(int m[SIZE], int *max1, int *max2)
{
   int i;
   *max1=m[0]>m[1]?m[0]:m[1];
   *max2=m[0]>m[1]?m[1]:m[0];
   /* missing code segment here */
}
```

(7 marks)

(c) The following program computes the sum of the elements of the two diagonals in a 3x3 two-dimensional array of integers. For example, if the array `a[3][3]={1,2,3,4,5,6,7,8,9}`, then `sum1` is 1+5+9=15, and `sum2` is 3+5+7=15. Write the missing code segment in the program. Note that you should not declare any other variables in the program.

```
#include <stdio.h>
int main()
{
   int a[3][3]={1,2,3,4,5,6,7,8,9};
   int i,j,sum1=0,sum2=0;
   /* missing code segment here */
   printf("sum1=%d,sum2=%d\n",sum1,sum2);
}
```

(10 marks)

3. (a) Stacks and queues are special types of list data structures. Describe how stacks and queues differ from regular linked list data structures.

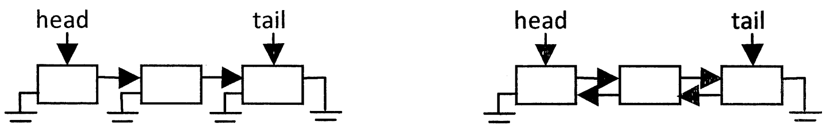
(3 marks)

(b) Describe the difference between stacks and queues.

(3 marks)

(c) Consider a singly-linked list that was built using doubly-linked list nodes. Write a C function `repairDblLinkedList()` that traverses such a linked list and repairs any broken links between nodes as depicted in Figure Q3. Pointers corresponding to broken links have been set to `NULL`. The function should return the number of broken links that were repaired. The function prototype is given as follows:

```
int repairDblLinkedList(DblLinkedList *ll);
```



(i) Before repair

(ii) After repair

Figure Q3

(12 marks)

(d) Consider the following representation of a polynomial of degree *n*, where a_i, *x* and *n* are all integer values:

$$a_n x^n + a_{n-1} x^{n-1} + a_{n-2} x^{n-2} + \ldots + a_1 x + a_0$$

Explain, using any necessary diagrams and/or C code, how you would use a linked list data structure to store a given polynomial.

(7 marks)

4. (a) Describe how binary search trees differ from general binary tree data structures.

(4 marks)

Note: Question No. 4 continues on Page 6

(b) Consider the binary tree T shown in Figure Q4. Use pre-order, in-order and post-order methods to traverse T. For each traversal order, do the following:

(i) State the print order of the values stored in each node.

(6 marks)

(ii) Show by drawing arrows on a copy of the diagram, the order in which nodes are visited during the tree traversal progress. Only draw an arrow when an actual node is visited.

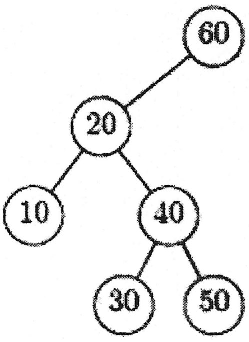


Figure Q4

(6 marks)

(c) Write a C function `printOddValDesc()` that accepts a pointer to the root node of a binary search tree B and prints any odd numbers stored in B in decreasing order of magnitude. The function prototype is given as follows:

```
int printOddValDesc(btnode *node);
```

(9 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.