浙江大学 2013－2014 学年夏季学期

《C 程序设计专题》课程期末考试试卷

课程号： 211Z0050 ， 开课学院： 计算机学院

考试试卷：√A 卷、B 卷（请在选定项上打√）

考试形式：√闭、开卷（请在选定项上打√），允许带 / 入场

考试日期： 2014 年 06 月 27 日,考试时间： 120 分钟

诚信考试，沉着应考，杜绝违纪.

考生姓名：______________学号：______________所属院系：____________

(注意：答题内容必须写在答题卷上，写在本试题卷上无效)

Section 1: single Choice(2 marks for each item, total 26 marks)

1. Suppose that five characters are pushed into a stack in the order of ***a, b, c, d, e***. The impossible popping sequence is ___.
   A. a b c d e　　B. e d c b a　　C. d c e a b　　D. d e c b a
2. When using a stack to evaluate the postfix expression ***2 3 5 + 6 * + 8 4 / +***, what symbols are inside the stack when "/" is read?___.
   A. +*+　　B. ++　　C. 2 3 5 6 8 4　　D. 50 8 4
3. To insert a node pointed by ***s*** after the node pointed by ***p*** in a linked list, we should do the following statements ___.
   A. s->next=p; p->next=s;　　B. s->next=p->next; p->next=s;
   C. s->next=p->next; p=s;　　D. p->next=s; s->next=p;
4. Assume that two structures are defined as following:
   struct data {
   　int day, month, year;
   };
   struct student {
   　char name[20];
   　long num;
   　struct data birthday;
   };
   And ten such structures are allocated:
   struct student *p = (struct student *)malloc(10 * sizeof(struct student));
   The expression ___ below can set the year of the second student's birthday to 1995 correctly.
   A. (p+1)->birthday.year = 1995;　　B. *(p+1)->birthday.year = 1995;
   C. (p+1)->birthday->year = 1995;　　D. (p+1).birthday->year = 1995;
5. About the function pointer, the statement ___ below is WRONG.
   A. The function pointer can be passed to a function as an actual argument.
   B. The function pointer can be used to call a function.
   C. Can't pass argument to a function when a function pointer is used to call the function.
   D. The function name is a pointer value which can be assigned to a function pointer variable.

6. About ***#include***, the statement___ below is correct.
   A. #include "xx.h" means that search the file xx.h in the current directory ONLY.
   B. #include <stdio.h> means that the ***stdio*** library is included in the program.
   C. #include <xx.h>means that search the file xx.h ONLY in the directory which be appointed by the compiler.
   D. #include<xx> is NOT correct because of missing the postfix ***.h***
7. Given the variables declaration ***char s[5]***;.If we convert the integer variable ***i*** to the corresponding character string, the statement ____ below is correct.
   A. sprintf(s, "%d", i);   B. sprintf(s, "%5d", i);
   C. sprintf(s, "%4d", i);   D. None of the above.
8. For the graphics library of our course, the statement ___ about the function ***printf*** below is correct.
   A. Printf can NOT output anything.
   B. Printf can output the content in the current position of the graphics window.
   C. Printf can open a text window and output the content automatically.
   D. If calling the function InitConsole(), printf will output the content in the special console window, otherwise nothing will be output.
9. Use ***typedef*** to define a function pointer type **PunPtr** which points to a function ***void func(int x)***. The correct expression is___.
   A. typedef FunPtr fun(int x);   B. typedef void FunPtr(int x);
   C. typedef void (*FunPtr)( );   D. typedef void (*FunPtr)(int);
10. The complexity(复杂度) of sort algorithm ___ below is O(N*Log$_2$N).
    A. Selection sort algorithm   B. Bubble sort algorithm
    C. Insertion sort algorithm   D. Merge sort algorithm
11. In the selection sort function ***SelectionSort(int a[],int n)***, a function ***swap*** is used to exchange two variables' values. The correct swap function which can fulfill(实现) the function(功能) is___.

```
A.  void swap(int a, int b) {           B.  void swap(int *a, int *b) {
            int t;                                  int *t;
            t=a;a=b;b=t;                            t=a;a=b;b=t;
    }                                       }
C.  void swap(int a[], int i, int j) {  D.  void swap(int *a, int *b) {
            int t;                                  int *t;
            t=a[i];a[i]=a[j];a[j]=t;               *t=*a;*a=*b;*b=*t;
    }                                       }
```

12. About the recursive function, the statement __ below is WRONG.
    A. The recursive function must call the function itself DIRECTLY.
    B. The recursion must have an EXIT(出口).
    C. The sub-problems in a recursive solution have the same form as the original problem.
    D. The C systems use the STACK strategy to implement the recursion .
13. Given the following definitions, the ***scanf*** statement ______ below is NOT correct.

```
struct pupil {
        char name[20];
        int age;
        int sex;
} pup[5], *p=pup;
```

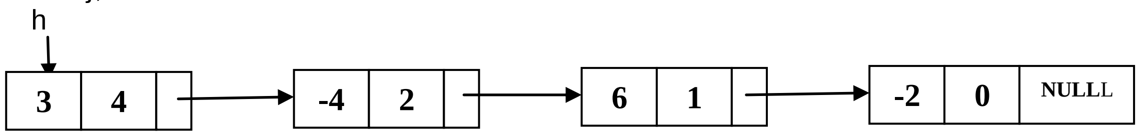
    A．scanf(“%s”,pup[0].name);   B．scanf(“%d”,&pup[0].age);
    C．scanf(“%d”, p->age);   D．scanf(“%d”,&(p->sex));

Section 2: Read the following problems and answer questions (5 marks for each item, total 30 marks)

1. Given the polynomial(多项式) definition of a linked list and a practical example (linked list ***h***), and the function ***f*** which is a mathematical operation(数学运算) on the

polynomial. Please draw up(画出) the result (a new polynomial) of the linked list ***h*** after calling the function ***f(h)***.

```
struct ListNode {
      int fact; /*coefficient(系数)*/
      int exp; /*exponent(指数)*/
      struct ListNode *next;
};
```



```
void f(struct ListNode *head)
{
     struct ListNode HEAD;
     struct ListNode *cur = head,  *pre = &HEAD;

     HEAD.next = head;
     If (!head) return;
     while(cur) {
          if(cur->exp) {
              cur->fact = cur->fact * cur->exp;
              cur->exp = cur->exp - 1;
          } else {
              free(cur);
              pre->next = NULL;
              break;
          }
          pre = cur;
          cur = cur->next;
     }
}
```

2. Given the definition of a linked list and a practical example (list ***h***), please write out the result of the linked list after calling the function ***f(h,3)***.

```
struct ListNode {
      int val;
      struct ListNode *next;
};
struct ListNode *f(struct ListNode *head, int x)
{
      struct ListNode *root = (struct ListNode *)malloc(sizeof(struct ListNode));
      struct ListNode *pivot = (struct ListNode *)malloc(sizeof(struct ListNode));
      struct ListNode *root_last = root;
      struct ListNode *pivot_last = pivot;
      struct ListNode *current = head;

      memset(root, 0, sizeof(struct ListNode));   /*Initial root spaces as zeroes*/
      memset(pivot, 0, sizeof(struct ListNode)); /* Initial pivot spaces as zeroes */
      while (current != NULL) {
          if (current->val < x) {
              root_last->next = current;
              root_last = current;
          } else {
              pivot_last->next = current;
              pivot_last = current;
          }
```

```
            current = current->next;
        }
        root_last->next = pivot->next;
        pivot_last->next = NULL;
        return root->next;
    }
```

The input linked list ***h*** is:
  1->4->3->2->5->2 and   x = 3.

3. The following program will output______________________.

```
#include <stdio.h>
int f(int x, int n, char s[])
{
   int count;

   if (x < 9) {
       s[n] = x + '0';
       s[n+1] = '\0';
       return n+1;
   }
   count = f(x/9, n, s);
   count = f(x%9, count, s);
   return count;
}
main()
{
    int a = 159;
    char s[100];

    f(a, 0, s);
    puts(s);
}
```

4. Write down the declaration of the function pointer variable ***p*** which points to a procedure(过程) ***f*** with two formal parameters: the first is an integer variable, the other is an integer pointer array.

5. Xiao Ming wrote a C program to call the library function ***printf("hello\n")*** in the main function , but he forgot to write down ***#include <stdio.h>*** in the beginning of the source file. However his program was be compiled and executed correctly. Why? Give a more example to explain that in what situation the program can be compiled but can NOT be executed correctly if the related head files are not be included.

6. Please describe ***the linear search algorithm***(线性查找算法) and ***the binary search algorithm***(二分查找算法). Evaluate their efficiencies(效率) and explain their use conditions(使用条件) respectively.

Section 3: According to the specification, complete each program (2 marks for each blank, total 24 marks)

1. Given a sorted linked list in ascending order（升序）, delete all nodes that have duplicate numbers, leaving only distinct numbers from the original list.

   For example,

   Given 1->2->3->3->4->4->5, return 1->2->5.

Given 1->1->1->2->3, return 2->3.

```
typedef struct _ListNode {
       int val;
       struct _ListNode *next;
} ListNode;

ListNode *insert(ListNode *target, ListNode *head, ListNode *tail)
   /*insert a node target after the tail of list which has a head node */
{
    if(!head->next)  /*Null list – only head node.*/
        head->next = target;   /*insert after head, as the first node */
    else
        tail->next = target;   /*insert after the tail */
    tail = target;
    tail->next = ___(1)____;
    return tail;
 }

 ListNode *deleteDuplicates(ListNode *head)
   /* head is the sorted linked list which has no head node */
 {
       ListNode result_head, *result_tail, *pre = head, *cur;

      if(!head || !head->next) return head;
      result_head.next = NULL;
      cur = head->next;
      while(pre) {
           while (cur && cur->val == pre->val)
                 cur = cur->next;
           if (cur ==____(2)____)
                result_tail = insert(pre, &result_head, result_tail);
           pre = cur;
           if (cur)  cur = cur->next;
     }
     return ____(3)____;
 }
```

2. Merge two sorted linked lists and return it as a new sorted list. The new list should be made by splicing(拼接) together the nodes of the first two lists. The sorted list is in ascending order.

```
typedef struct _ListNode {
       int val;
       struct _ListNode *next;
} ListNode;

 ListNode *mergeTwoLists(ListNode *l1, ListNode *l2)
 {
    ListNode HEAD, *p;
    ListNode *cur = &HEAD;

    while (l1 && l2) {
        if(___(4)____) {
            cur->next = l1;
            l1= l1->next;
        } else {
            cur->next = l2;
            l2 = ___(5)___;
        }
        cur = cur->next;
    }
    p = ___(6)___;
    cur->next = p;
    return HEAD.next;
}
```

3. The graphics library of our course uses a timer(定时器) strategy which contains three functions：
   - void RegisterTimerEvent(TimerEventCallback callback); /*register the callback function*/
   - void starttimer(int timerID, int timeinterval);   /*start the timer*/
   - void canceltimer(int timerID);   /*close the timer*/

   Now we wants to implement a single timer task(单次定时任务) that outputs a string ***"Hello World"*** in the console window after 1000 ms, then the display will not occur again. Please complete the following program fragment.
   NOTE: If you think that some places don't need to be filled in anything, you can leave them blanks. Don't care about the codes which create the window.

```
void timerCallBack(int timeID)
{
    printf("Hello World\n");
    ___(7)___;
}
……
void Main()
{
    .....
    RegisterTimerEvent(___(8)___);
    starttimer(___(9)___);
    .....
}
```

4. The following program uses the recursive quick sort algorithm to sort the character strings in ascending order（升序）according to the compare strategy in the function ***numcmp***. The program will output:

```
    3
    22
    111
```

   Complete the program.

```
#include <stdio.h>
#include <stdlib.h>

/* numcmp: compare s1 and s2 numerically */
int numcmp(char *s1, char *s2)
{
    int v1, v2;

    v1 = atoi(s1);
    v2 = atoi(s2);
    if (v1 < v2)
        return -1;
    else if (v1 > v2)
```

```
            return 1;
        else
            return 0;
    }

    void swap(char *v[], int i, int j)
    {
          char *temp;

           temp = v[i];
           v[i] = v[j];
           v[j] = temp;
    }

    /* qsort: sort v[left]...v[right] into increasing order */
    void qsort(char *v[], int left, int right, int (*comp)(char *, char *))
    {
           int i, last;
           void swap(char *v[], int, int);

           if (left >= right)         /* do nothing if array contains nothing*/
               return;
           swap(v, left, (left + right)/2);
           last = left;
           for (i = left+1; i <= right; i++)
                if((*comp)(v[i], v[left]) < 0)
                    swap(v, ++last, i);
           swap(v, left, last);
           qsort(v, ______(10)______________);
           qsort(v, ______(11)______________);
    }

    main()
    {
        char *str[]={"111","22","3"};
        int i;

        qsort( _________(12)_____________);
        for(i=0;i<3;i++)   printf("%s\n",str[i]);
    }
```

Section 4: Algorithms design (10 marks for each item, total 20 marks)

1. Given a linked list (no cycle in the list), remove the ***nth*** node from the end of list and return its head. Please (1) describe your algorithm of this problem, and (2) give your function ***ListNode *removeNthFromEnd(ListNode *head, int n).***

For example,

Given linked list: 1->2->3->4->5, and n = 2.

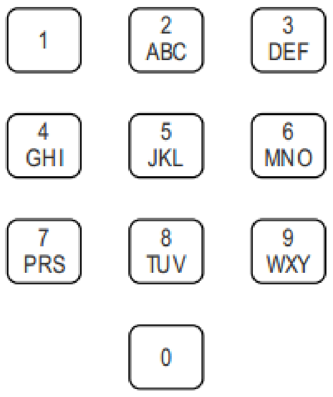
After removing the second node from the end, the linked list becomes 1->2->3->5.

You must travel the linked list only once.

Definition of the linked list:

```
typedef struct _ListNode {
        int val;
        struct _ListNode *next;
} ListNode;
```

2. On a standard telephone keypad, the digits are mapped onto the alphabet (minus the letters Q and Z) as shown in this diagram.



In order to make their phone numbers more memorable, service providers like to find numbers that spell out some word appropriate to their business that makes that phone number easier to remember. Such words that help you remember some other data are called ***mnemonics***.
In the following program, The function ***ListMnemonics*** will generate all possible letter combinations that correspond to a given number, represented as a string of digits.
Write down the definition of recursive function ***ListMnemonics*** to complete the program.

```
#include <stdio.h>
#include "genlib.h"
#include "strlib.h"
#include "simpio.h"

/* Private function prototypes */
static void ListMnemonics(string str);

/*data structure*/
static string dialpad[] = { /*map table from dial-pad to their letters */
            "0", "1", "ABC", "DEF", "GHI", "JKL", "MNO", "PRS", "TUV", "WXY"
};
static char Mnemonics[100] = {'\0'}; /*Store the mnemonics string */
static int index = 0;   /*Record the current letter position of the mnemonics */

main()
{
    string str;
    printf("This program lists all Mnemonics of a telephone number.\n");
    printf("Enter a telephone number string: ");
    str = GetLine();
    ListMnemonics(str);
}

static void ListMnemonics(string str)
{
      ……
}
```