浙江大学 2017－2018 学年夏学期

《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：__2018__年__07__月__05__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 20 marks)

1. Given the definitions as below:

```
typedef struct {double x, y;} PointT;
PointT p1, p2;
int i;
char *pc1 = (char *)&p1, *pc2 = (char *)&p2;
```

In the followings, the statement _____ ***CANNOT*** copy the value of ***p1*** to ***p2*** CORRECTLY.

A. p2 = p1;
B. p2.x = p1.x; p2.y = p1.y;
C. strcpy(pc2, pc1);
D. for (i = 0; i < 16; i++) *pc2++ = *pc1++;

2. After executing the following code fragment, the value of variable ***z*** is _____.

```
static struct{
    int x, y[3];
} a[3] = {{1,2,3,4},{5,6,7,8},{9,10,11,12}}, *p = a+1;
int z;
z = ((int *)p+1)[-2];
```

A. 4　　B. 5　　C. 6　　D. 7

3. Six elements ***a,b,c,d,e,f*** are pushed into a stack consequently, and pop operations are performed alternately. If there are ***NO*** continuous three pop operations(连续 3 次 pop 操作), which of the followings is the impossible output? _____

A. afedcb　　B. cbdaef　　C. dcebfa　　D. bcaefd

4. In a linked list, which of the following is the possible statement to delete the node after ***p***? _____

A．p=p->next;　　B．p=p->next->next;
C．p->next=p;　　D．p->next=p->next->next;

5. For the following declarations, which ***scanf*** statement is wrong? _____

```
struct Student{
    char name[10];
    int age;
    char gender;
```

} std[3], *p=std;

A. scanf("%d",&(*p).age); B. scanf("%c",&std[0].gender);
C. scanf("%c",&(p->gender)); D. scanf("%s",&std.name);

6. What is the output of the following statements? _____

```
struct {
    int x, y;
} s[2] = { { 1, 3 }, { 2, 7 } };
printf("%d", s[0].y/s[1].x );
```

A. 0 B. 1 C. 2 D. 3

7. when sorting ***n*** objects, if input array is already sorted, the Bubble Sort algorithm has _____ time complexity.
A. O(1) B. O(nlogn) C. O(n) D. O(n^2)
8. Inserting a node into a descending-order(降序) linked list with ***n*** nodes needs _____ comparisons at average.
A. O(1) B. O(nlogn) C. O(n^2) D. O(n)
9. For keyword ***static***, which one below is NOT correct? _____
A. A static local variable is persistence after calling the function.
B. A static global variable is not accessible by other compile units.
C. A static function is not accessible by other compile units.
D. "static int *p;"means that p is a pointer to a static int.
10. According to the following function-declaration, the return value is _____ after calling ***f(4)***.

```
int f(int n)
{  if (n)  return f(n - 1) + n;  else  return n;  }
```

A. 0 B. 4 C. 10 D. None of the above

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

1. Write down your answers.
(1) When calling ***fun("2018-07-05");*** the output is _____.

```
void fun(char *str)
{
   if (*str) fun(str+1);
   if (*str) putchar(*str);
   return;
}
```

(2) Given a function ***fun*** which has a formal parameter of type ***int*** and returns a pointer to the function ***void f(int n)***. How to describe the prototype of ***fun***?
_____

2. When input ***23<ENTER>***, the following program will output_____.

```
#include <stdio.h>
#include <string.h>
char *dialpad[] = {
     "0","1","AB","CDE","FGH","JKL","MNO","PRS","TUV","WXY"
};
char Mnemonics[20];
int indx = 0;

int main()
{
     void ListMnemonics(char *str);
     char num[10];
     scanf("%s", num);
     ListMnemonics(num);
}
```

```
void ListMnemonics(char *str)
{
     char *ptr;
     if (strlen(str)==0) {
        Mnemonics[indx] = '\0';
        printf("%s#", Mnemonics);
     } else {
        ptr = dialpad[*str-'0'];
        while (*ptr) {
              Mnemonics[indx++] = *ptr++;
              ListMnemonics(str+1);
        }
   }
   return;
}
```

3. Execute the fllowing program, what will happen when press the key ***ESC***? _____
Note: the following program is based on our course's graphics library.

```
#include <windows.h>
#include <winuser.h>
#include "graphics.h"
#include "extgraph.h"
#include "genlib.h"
bool flag = FALSE;
double x, y;
void KeyboardEventProcess(int key,int event);
void TimerEventProcess(int timerID);

void Main()
{
   InitGraphics();
   registerKeyboardEvent(KeyboardEventProcess);
   registerTimerEvent(TimerEventProcess);
   x = GetWindowWidth()/2;
   y = GetWindowHeight()/2;
   return;
}

void KeyboardEventProcess(int key,int event)
{
   if (event==KEY_DOWN && key==VK_ESCAPE) startTimer(1, 500);
   return;
}

void TimerEventProcess(int timerID)
{
   static char buf[2] = "9";
   static int count = 0;
   static flag = TRUE;
   if (timerID == 1) {
      SetEraseMode(!flag);
      MovePen(x, y);
      DrawTextString(buf);
      if (flag = !flag) buf[0]--;
      if (++count == 20) cancelTimer(1);
   }
   return;
}
```

4. For the following program and a linked list ***h***: ***1->2->3->4->5->6->7->8***,
(1) what will the function-call ***f(h,3)*** prints out? _____
(2) what will the function-call ***f(h,3)*** return? _____

```
struct ListNode {
    int data;
    struct ListNode *link;
} ;
typedef struct ListNode *LinkList
LinkList h;

int f( LinkList list, int k )
{
    LinkList p, q;
    int count = 0;
    p = q = list->link;
    while ( p && count<k ) {
        p = p->link;
        count++;
    }
    if ( p ) {
       while ( p ) {
           p = p->link;
           q = q->link;
       }
    }
    if ( count < k )
        return 0;
    else {
        printf("%d\n",q->data);
        return 1;
    }
}
```

5. For the following program and a linked list ***h***: ***(-3)->(4)->(5)->(-2)->(-1)->(2)->(6)->(-2)***, after executing process(***h***), what will the program output?

```
struct node {
      int data;
      struct node *next;
};
void process(struct node *h)
{
      struct node *p, *thisp,*maxp;
      int this=0, max=0;
      p=thisp=h;  maxp=NULL;
      while (p) {
          this=this+p->data;
          if (this>max) {
              max=this;
              maxp=thisp;
          } else if (this<0) {
               this=0;
               thisp=p->next;
          }
          p=p->next;
      }
      p=maxp;
```

```
        while (p && (max>0)) {
            printf("%d " , p->data);
            max=max - p->data;
            p=p->next;
        }
        return;
    }
```

6. Gnome Sort also called Stupid sort is based on the concept of a Garden Gnome sorting his flower pots(花盆). A garden gnome sorts the flower pots by the following method:
   (1) He looks at the flower pot next to him and the previous one; if they are in the right order he steps one pot forward, otherwise he swaps them and steps one pot backwards.
   (2) If there is no previous pot (he is at the starting of the pot line), he steps forwards; if there is no pot next to him (he is at the end of the pot line), he is done.

   We can implement Gnome Sort as follows.

```
void gnome(int arr[], int n)
{
    int i = 0, t;
    while (i < n) {
        if (i == 0 || arr[i] >= arr[i-1]) {
            i++;
        } else {
            t = arr[i-1];
            arr[i-1] = arr[i];
            arr[i] = t;
            i--;
        }
    }
    return;
}
```

   Please describe one of the best case for this algorithm and provide the best-case performance (computational complexity).

Section 3: According to the specification, complete each program (2 marks for each blank, total 30 marks)

1. The following program realizes the function of strokes, that is, when the left mouse button is pressed and dragged, a trace is drawn in the window along with the mouse position, and when the left mouse button is raised, the trace is not drawn. Please fill in the blanks to complete the program.
   Note: the following program is based on our course's graphics library.

```
#include <windows.h>
#include <winuser.h>
#include "graphics.h"
#include "extgraph.h"
#include "genlib.h"

void Painter(int x, int y, int button, ____(1)____ )
{
    double cx = ScaleXInches(x);
    double cy = ScaleYInches(y);
    double dx,dy;
```

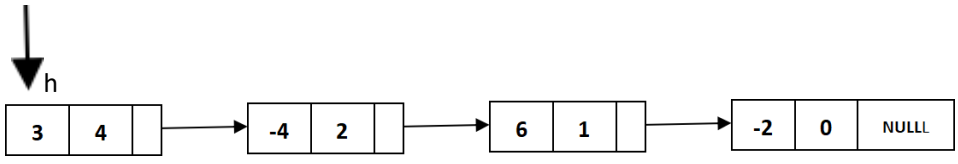
```
    static int isLeftButtonDown = 0;
    ____(2)___ lx = 0.0, ly = 0.0;

    switch(event){
       case BUTTON_DOWN:
           if (button==LEFT_BUTTON)   isLeftButtonDown = 1;
           lx = cx;    ly = cy;
           break;
        case BUTTON_UP:
           if (button==LEFT_BUTTON)  isLeftButtonDown = 0;
           break;
       case ____(3)____:
          if (isLeftButtonDown) {
              MovePen(lx,ly);
              dx = cx-lx; dy = cy-ly;
               DrawLine(dx,dy);
               lx = cx; ly = cy;
          }
          break;
    }
    return;
}

void Main()
{
    _____(4)_____;
    SetPenSize(1);
    SetPenColor("Black");
    _____(5)_____;
    return;
}
```

2. The following linked list is a polynomial(多项式), where each node is a term of the polynomial consisting of coe (coefficient) and exp (exponential). The following function f(h,x) is to compute the value of the polynomial h with a certain x.

```
   struct node {
         int coe;
         int exp;
         struct node *next;
    } ;
    typedef  struct node  ListNode;
```




```
 int f(ListNode *h, int x)
 {
       ____(6)____ *p=h;
       int res=0, last, cur, i;

       if (h==NULL) return res;
       last=h->exp;

       while(p!=NULL) {
```

```
            cur=p->exp;
            for (i=last; i>cur; i--)
                 res=      (7)      ;
            res += _____(8)_____;
             last=cur;
             p=_____(9)_____;
        }
        for (i=last; i>0; i--) res=res*x;
        _____(10)_____;
    }
```

3. The Binary Insertion Sort algorithm use binary search to find the proper location to insert the selected item at each iteration, thus, it reduces the number of comparisons in normal insertion sort. The final output of our implementation should be as follows.

   0 12 17 23 31 37 46 54 72 88 100

   Please complete the following Binary Insertion Sort implementation.

```
#include <stdio.h>
int binarySearch(int a[], int item, int low, int high)
{
    int mid;

    if (high <= low) return (item > a[low])?  (low + 1): low;
    mid = _____(11)_____;
    if(item == a[mid])  return mid;
    if(item > a[mid])  return binarySearch(a, item, mid+1, high);
    return binarySearch(a, item, low, ____(12)____);
}

void insertionSort(int a[], int n)
{
    int i, loc, j, k, selected;

    for (i = 1; i < n; ++i)  {
        j = i - 1;
        selected = a[i];
        loc = binarySearch(a, selected, 0, ____(13)____);
        while (j >= loc) {
            a[j+1] = a[j];
            j--;
        }
        a[j+1] = _____(14)_____;
    }
    return;
}

int main()
{
    int a[] = {37, 23, 0, 17, 12, 72, 31, 46, 100, 88, 54};
    int n = __(15)__/sizeof(a[0]) , i;

    insertionSort(a, n);
    for (i = 0; i < n; i++) printf("%d ",a[i]);
    return 0;
}
```

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

1. For a linked list ***h***, function ***int fun(struct node *h)*** is to determine whether the linked list ***h*** is central symmetry（中心对称）. If it is, return 1, or return 0. E.g. ***1->2->3->2->1*** and ***1->2->2->1*** is central symmetry, and ***1->2->1->2*** is not. Please implement the function ***fun(struct node *h)*** with a ***STACK***. You need not to implement the function ***push()*** and ***pop()***. You can call the functions directly: ***void push(int n), int pop()***.

```
struct node {
      int data;
      struct node *next;
 };

 int process(struct node *h)
 {
      ……
 }
```

2. Given an array of numbers, push all the zeros to the end of the array. For example, if the given arrays is ***{1, 9, 8, 4, 0, 0, 2, 7, 0, 6, 0, 9}***, it should be changed to ***{1, 9, 8, 4, 2, 7, 6, 9, 0, 0, 0, 0}***. The order of all other elements should be same. Expected time complexity is ***O(n)*** and extra space is ***O(1)***.

   Assuming we already implement the main function as follows.

```
int main()
{
   int arr[] = {1, 9, 8, 4, 0, 0, 2, 7, 0, 6, 0, 9};
   int n = sizeof(arr) / sizeof(arr[0]);
   void push_zeros_end(int arr[], int n);

   push_zeros_end(arr, n);
   for (int i = 0; i < n; i++) printf("%d ", arr[i]);
   return 0;
}
```

   Please implement the ***push_zeros_end*** function.

```
void push_zeros_end(int arr[], int n)
{
      ……
}
```