浙江大学 2018－2019 学年夏学期

《C 程序设计专题》课程期末考试题参考答案

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： 2019 年 06 月 26 日，考试时间： 120 分钟

试题号	一	二	三	四	总分	
满分	20	30	30	20		
得分					统分人 1	
阅卷人					统分人 2	

Section 1: Single Choice(2 marks for each item, total 20 marks)

1 A	2 C	3 C	4 A	5 D
6 C	7 C	8 B/D	9 D	10 C

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

1. (1) Yes. Because vector is not a variable (vector is a type name). (2’)

   (2) typedef int *(*T)(char *, double); (3’)

2. 36 4 20 4

3. 2->3->7->4

4. 1->2->3->4->5->6

5. 3#Empty2!0#2#1#

6. 7

院系________ 学号________ 姓名________ 任课教师________

装 订 线

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

(1) mAB

(2) DrawTriangle(A,B,C)

(3) FraTriangle(A,mAB,mCA)

(4) FraTriangle(mAB,B,mBC)

(5) FraTriangle(mCA,mBC,C)

(6) Q->front==(Q->rear+1)%Q->maxsize

(7) Q->front==Q->rear

(8) Q->pBase[Q->rear]

(9) (Q->rear+1)%Q->maxsize

(10) *val

(11) InitGraphics()

(12) KeyboardEventProcess

(13) TimerEventProcess

(14) TIMER_BLINK100 ***or*** 1

(15) isDisplayCircle=!isDisplayCircle

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

1.

```
static ListNode* FindFirstCommonNode(
ListNode* l1, ListNode* l2) {
  int len1, len2, numLeftNodes;
  ListNode *lPtr, *sPtr;
if (l1==NULL || l2==NULL) return NULL;
  len1 = ListLength(l1);
  len2 = ListLength(l2);
  if (len1 > len2) {
    lPtr = l1; sPtr = l2;
    numLeftNodes = len1-len2;
  } else {
    lPtr = l2; sPtr = l1;
    numLeftNodes = len2-len1;
  }
  for (int i=0;i<numLeftNodes;i++)
    lPtr = lPtr->next;
  while (lPtr && sPtr && lPtr!=sPtr) {
    lPtr = lPtr->next;
    sPtr = sPtr->next;
  }
  return lPtr;
}
```

2.

```
void binSelection(int array[], int n) {
  int k,tmp,lh,rh,minPos,maxPos;
  for (lh=0, rh=n-1; lh<rh; lh++, rh--) {
    minPos=lh; maxPos=rh;
    for (k=lh; k<=rh; k++) {
      if (array[minPos]>array[k]) {
        minPos = k;
      } else if (array[maxPos]<array[k]) {
        maxPos=k;
      }
    }
    tmp=array[lh];
    array[lh]=array[minPos];
    array[minPos]=tmp;
    tmp=array[rh];
    array[rh]=array[maxPos];
    array[maxPos]=tmp;
  }
  return;
}
```