2021年C语言期末试题-参考

注：非官方答案，不保证正确，仅供参考!!

一、写出下列程序段的运行结果（40分）

自行在IDE中验证，不懂可以输出中间结果

二、改错题（20分）

本题按常规应该有10~11个错误，因此以下答案很可能遗漏了，有找到的同学可以反映

3行：#define SIZE 50; -> #define SIZE 50

9行：load函数只有定义未声明，添加void load(struct Goods a[]);

14行：int flag; -> int flag = 0;

16行：load(a[SIZE]); -> load(a);

17行：scanf ("%s", &strtemp); -> scanf ("%s", strtemp);

20行：if (strtemp == a[i].item) -> if (strcmp(strtemp, a[i].item) == 0) 或 if (!strcmp(strtemp, a[i].item))

23行：a[i].price+=1.1; -> a[i].price*=1.1;

31行：return; -> return 0;

37行：if ((fp = fopen ("d:\data.dat", "rb")) == NULL) -> if ((fp = fopen ("d:\\data.dat", "rb")) == NULL)

43行：if (fread(a[i], sizeof(struct Goods), 1, fp) != 1) -> if (fread(&a[i], sizeof(struct Goods), 1, fp) != 1)

三、编程题（40分）

1. （12分）

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

int main(){
    int n, k, sum = 0;
    char *num = calloc(200, sizeof(char));
    char **nums = calloc(100, sizeof(char*));
```

```
   for(int i = 0; i < 100; i++){
      nums[i] = calloc(20, sizeof(char));
   }

   scanf("%d %d", &n, &k);
   scanf("%s", num);

   int n_num = n / k + ((n % k) ? 1 : 0);
   int len = (n % k) ? (n % k) : k;
   int flag = (n_num % 2) ? 1 : -1;
   for(int i = 0; i < n_num; i++){
      strncpy(nums[i], num, len);
      nums[i][len] = '\0';
      if (flag < 0){
         strrev(nums[i]);
      }
      sum += atoi(nums[i]);
      flag = -flag;
      num = num + len;
      len = k;
   }

// for(int i = 0; i < n_num; i++){
//    printf("%s\n", nums[i]);
// }

   printf("%d\n", sum % (int)pow(10, k));
   return 0;
}
```

2. （14 分）

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

#define MAX_GUESTS 20
#define NAME_LENGTH 32

typedef struct {
    char name[NAME_LENGTH];
    int height;
} Guest;

// 进行比较，身高相同时按字典序排
int compare(const Guest *a, const Guest *b) {
    if (a->height != b->height) {
        return a->height - b->height;
    }
    return strcmp(a->name, b->name);
}

// 第一次排序，从矮到高
void sort1(Guest guests[], int n) {
    int i, j;
    for (i = 0; i < n - 1; i++) {
        for (j = 0; j < n - 1 - i; j++) {
            if (compare(&guests[j], &guests[j + 1]) > 0) {
                Guest temp = guests[j];
                guests[j] = guests[j + 1];
                guests[j + 1] = temp;
            }
        }
    }
}

// 第二次排序，身高高的在中间
void sort2(Guest guests_old[], Guest guests_new[], int n){
    int pos = n / 2;
    int direction = (n % 2 - 0.5) * 2;
    int distance = 1;
    for(int i = n - 1; i >= 0; i--){
        guests_new[pos] = guests_old[i];
        pos += direction * distance++;
        direction = -direction;
    }
}

void printGuests(const Guest guests[], int n) {
    int i;
    for (i = 0; i < n; i++) {
        printf("(%s %d)", guests[i].name, guests[i].height);
    }
    printf("\n");
}

int main() {
    Guest guests[MAX_GUESTS];
    Guest guests_res[MAX_GUESTS];

    int n, i;
    scanf("%d", &n);
    for (i = 0; i < n; i++) {
        scanf("%s %d", guests[i].name, &guests[i].height);
    }

    sort1(guests, n);
    sort2(guests, guests_res, n);
    printGuests(guests_res, n);

    return 0;
}
```

3. （14分）

（1）（2分）

```
typedef struct N {
    char s[81]; // Assuming string length max 80 + 1 for '\0'
    struct N *next;
} Node;
```

（2）（5分）

```
int t(char *s) {
    if (s == NULL) {
        return 0;
    }

    int freq[128] = {0};
    char minChar = 127;

    for (int i = 0; s[i] != '\0'; ++i) {
        freq[(int)s[i]]++;
        if (s[i] < minChar) {
            minChar = s[i];
        }
    }

    return freq[(int)minChar];
}
```

（3）（7分）

```
Node* ex(Node *A, int n) {
    Node *B = NULL;
    Node *tail = NULL;

    for (Node *current = A; current != NULL; current = current->next) {
        if (t(current->s) == n) {
            Node *newNode = (Node *)malloc(sizeof(Node));
            if (newNode == NULL) {
                fprintf(stderr, "Memory allocation failed.\n");
                exit(EXIT_FAILURE);
            }

            strcpy(newNode->s, current->s);
            newNode->next = NULL;

            if (B == NULL) {
                B = newNode;
            } else {
                tail->next = newNode;
            }
            tail = newNode;
        }
    }

    return B;
}
```