Experiment11-董皓彧

环境：

```
gcc.exe (x86_64-win32-seh-rev0, Built by MinGW-W64 project) 8.1.0
Visual Stdio Code 1.84.2
```

作业仓库地址：
https://github.com/FHYQ-Dong/Tsinghua-Program-Design-Assignments/tree/main/Experiment11

必做题

Experiment11-1

题目：

> 编写程序，将两个长度各为 10 的整型数组数据按从小到大排序，然后将两组数据合并到一个长度为 20 的整型数组中，合并后的数组仍然按照从小到大排序。

输入格式：

> 共 2 行，分别为两个数组

输出格式：

> 共 1 行，为合并后的数组

代码：

```c
#include <stdio.h>
typedef char bool;
#define true 1
#define false 0

void swap(int *a, int *b) { int t = *a; *a = *b; *b = t; }

void qsort(int* begin, int *end, bool (*cmp)(int, int)) {
    if (begin >= end) return;
    int *l = begin, *r = end-1, *p = begin;
    while (l < r) {
        while (l < r && cmp(*p, *r)) --r;
        while (l < r && cmp(*l, *p)) ++l;
        if (l < r) swap(l, r);
    }
    swap(p, l);
    qsort(begin, l, cmp);
    qsort(l+1, end, cmp);
    return;
}
```

```
bool cmp(int a, int b) { return a <= b; }

void merge(int* source1, int* source2, int* dest) {
    int *p1 = source1, *p2 = source2, *p = dest;
    while (p1 < source1+10 && p2 < source2+10) {
        if (*p1 < *p2) *(p++) = *(p1++);
        else *(p++) = *(p2++);
    }
    while (p1 < source1+10) *(p++) = *(p1++);
    while (p2 < source2+10) *(p++) = *(p2++);
    return;
}

int main() {
    int a[10], b[10], res[20];
    for (int i=0; i<10; ++i) scanf("%d", &a[i]);
    for (int i=0; i<10; ++i) scanf("%d", &b[i]);
    qsort(a, a+10, cmp); qsort(b, b+10, cmp);
    merge(a, b, res);
    for (int i=0; i<20; ++i) printf("%d ", res[i]);
    return 0;
}
```

输入1：

```
1 12 3 14 5 16 7 18 9 20
10 19 8 17 6 15 4 13 2 11
```

输出1：

```
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
```

输入2：

```
1 3 5 7 9 11 13 15 17 19
2 4 6 8 10 12 14 16 18 20
```

输出2：

```
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
```

输入3：

```
1 1 1 1 1 1 1 1 1 1
2 2 2 2 2 2 2 2 2 2
```

输出3：

```
1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2
```

输入4：

```
1 2 3 4 5 6 7 8 9 10
10 9 8 7 6 5 4 3 2 1
```

输出4：

```
1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8 9 9 10 10
```

输入5：

```
1 2 3 4 5 6 7 8 9 10
1 2 3 4 5 6 7 8 9 10
```

输出5：

```
1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8 9 9 10 10
```

Experiment11-2

题目：

```
有 n（n<50）个人围成一圈，顺序编号。从第 1 个人开始报数（1，2，3），报数为 3 者退出圈子。问最后留下来的人刚开始时排在几号?
```

输入格式：

```
共 1 行，一个整数 n
```

输出格式：

```
共 1 行，一个整数，表示最后留下来的人刚开始时排在几号
```

代码：

```
#include <stdio.h>
#include <string.h>
#define scan(x) scanf("%d", &(x))
#define print(x) printf("%d ", x)
#define endl putchar('\n')

int del(int n, int* people) {
    int cnt = 1, idx = 1;
    while (people[idx] != idx) {
        if (cnt == 2) {
            people[idx] = people[people[idx]];
            idx = people[idx];
            cnt = 1;
        }
        else {
            ++cnt;
            idx = people[idx];
        }
```

```
    }
    return idx;
}

int main() {
    int people[51], n;
    scan(n);
    for (int i=1; i<=n; ++i) people[i] = i % n + 1;
    int res = del(n, people);
    print(res);
    return 0;
}
```

输入1：

```
10
```

输出1：

```
4
```

输入2：

```
1
```

输出2：

```
1
```

输入3：

```
50
```

输出3：

```
11
```

输入4：

```
2
```

输出4：

```
2
```

输入5：

```
29
```

输出5：

```
26
```

Experiment11-3

题目：

> 编写程序，从键盘读入 10 个整数，将其存在一个长度为 10 的一维数组 a[] 中。然后输出该组数据从小到大的排序结果以及在原数组中的下标。（假设输入中无重复数据）

输入格式：

```
共 1 行，10 个整数
```

输出格式：

```
共 2 行：
第 1 行，排序后的数组；
第 2 行，在原数组中的下标。
```

代码：

```
#include <stdio.h>
typedef char bool;
typedef int* intp;
#define true 1
#define false 0

void bsort(intp* begin, intp* end, bool (*cmp)(intp, intp)) {
    if (begin >= end) return;
    for (intp* i = begin; i < end; ++i) {
        for (intp* j = i+1; j < end; ++j) {
            if (cmp(*j, *i)) {
                intp t = *i;
                *i = *j;
                *j = t;
            }
        }
    }
    return;
}

bool cmp(intp a, intp b) { return *a < *b; }

int main() {
    int a[10];
    intp pa[10];
    for (int i=0; i<10; ++i) scanf("%d", &a[i]), pa[i] = &a[i];
    bsort(pa, pa+10, cmp);
    for (int i=0; i<10; ++i) printf("%d ", *pa[i]);
    printf("\n");
    for (int i=0; i<10; ++i) printf("%d ", pa[i]-a);
}
```

输入1：

```
10 9 8 7 6 5 4 3 2 1
```

输出1：

```
1 2 3 4 5 6 7 8 9 10
9 8 7 6 5 4 3 2 1 0
```

输入2：

```
1 10 2 9 3 8 4 7 5 6
```

输出2：

```
1 2 3 4 5 6 7 8 9 10
0 2 4 6 8 9 7 5 3 1
```

输入3：

```
6 7 8 9 10 5 4 3 2 1
```

输出3：

```
1 2 3 4 5 6 7 8 9 10
9 8 7 6 5 0 1 2 3 4
```

输入4：

```
1 2 3 4 5 6 7 8 9 10
```

输出4：

```
1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9
```

输入5：

```
6 3 7 8 2 5 10 1 9 4
```

输出5：

```
1 2 3 4 5 6 7 8 9 10
7 4 1 9 5 0 2 3 8 6
```

选做题

Optional-Experiment11-1

题目：

> 下列程序的输出结果是?

输入格式：

> 无

输出格式：

> 输出结果，具体解析见代码注释

代码：

```c
#include <stdio.h>

char *a = "HAPPYNEWYEAR";
char b[] = "happynewyear";

int main() {
    int i = 8;
    printf("%c%c%s%s\n", *a, b[0], b+5, &a[5]);
    /*
        * 输出: HhnewyearNEWYEAR
        * %c ---> *a : char *a 的第一个字符
        * %c ---> b[0] : char b[] 的第一个字符
        * %s ---> b+5 : char b[] 的第六个字符开始的字符串
        * %s ---> &a[5] : char *a 的第六个字符开始的字符串
    */
    while (*(a+i)) putchar(*(a+i++));
    /*
        * 输出: NEWYEAR
        * 因为 char *a 的结尾字符为 '\0', 所以 *(a+i) 为 '\0' 时, 循环结束。
        * i 初始值为 8, *(a+i++) 依次为 a[8], a[9], ...
    */
    return 0;
}
```

Optional-Experiment11-2

题目：

> 验证卡布列克运算。任意一个四位数，只要它们各个位上的数字是不全相同的，就有这样的规律：
> 1. 将组成这个四位数的 4 个数字由大到小排列，形成由这 4 个数字构成的最大的四位数；
> 2. 将组成这个四位数的 4 个数字由小到大排列，形成由这 4 个数字构成的最小的四位数（如果4个数字中含有0，则得到的数不足四位）；
> 3. 求两个数的差，得到一个新的四位数；
> 4. 重复以上过程，最后得到的结果总是 6174。

输入格式：

共 1 行，一个整数 N

输出格式：

若 N 为各个位上数字不全相同的四位数，则输出验证卡布列克运算的公式；否则输出 "error"。

代码：

```
#include <stdio.h>

void swap(int *a, int *b) { int t = *a; *a = *b; *b = t; }

void kablek(int num) {
    int digit[4];
    for (int i=0; i<4; ++i) {
        digit[i] = num % 10;
        num /= 10;
    }
    for (int i=0; i<4; ++i) for (int j=0; j<4; ++j) if (digit[j]>digit[i])
swap(&digit[i], &digit[j]);
    int min_num = 1000*digit[0] + 100*digit[1] + 10*digit[2] + digit[3], max_num
= 1000*digit[3] + 100*digit[2] + 10*digit[1] + digit[0];
    int diff = max_num - min_num;
    printf("%d-%d=%d\n", max_num, min_num, diff);

    if (diff == 6174) return;
    else kablek(diff);
    return;
}

int main() {
    int a; scanf("%d", &a);
    if (a < 1000 || a > 9999) {
        printf("error\n");
        return 0;
    }
    if (a % 1111 == 0) {
        printf("error\n");
        return 0;
    }
    kablek(a);
    return 0;
}
```

输入1：

```
1234
```

输出1：

```
4321-1234=3087
8730-378=8352
8532-2358=6174
```

输入2：

```
1234567
```

输出2：

```
error
```

输入3：

```
9999
```

输出3：

```
error
```

输入4：

```
9998
```

输出4：

```
9998-8999=999
9990-999=8991
9981-1899=8082
8820-288=8532
8532-2358=6174
```

输入5：

```
1000
```

输出5：

```
1000-1=999
9990-999=8991
9981-1899=8082
8820-288=8532
8532-2358=6174
```