Задание

1. Разработать алгоритм решения задачи (см. варианты ниже)

2. Написать функцию для решения задачи и основную программу.

3. Подобрать тестовые данные. Отладить программу.

Ограничения на алгоритм и программу

1. Глобальные параметры не использовать.

2. Динамическую память не использовать.

3. Все массивы должны быть одного типа. В массивах первые элементы содержат полезную информацию (заданные числа или результат решения задачи), в остальных – мусор. Для каждого массива должна быть переменная, которая хранит количество элементов массива с полезной информацией.

4. В функции не использовать дополнительные массивы и другие структуры данных.

5. Функция не должна вызывать другие функции.

6. Исходные данные (массивы A и B) вводить в основной программе, а не в функции для решения задачи.

7. Результат задачи (массив C) выводить в основной программе, а не в функции для решения задачи.

8. В решении вариантов 8–12 нельзя использовать вложенные циклы, а также цикл, моделирующий вложенный цикл.

Вариант 1

Дано:

A – массив натуральных чисел, в котором нет одинаковых элементов;

B – массив натуральных чисел, в котором нет одинаковых элементов.

Получить массив C, содержащий все элементы массивов A и B без повторений.

Решение:

```c
#include <stdio.h>
#include <windows.h>

void inputArray(int *a, const size_t n) {
    for (size_t i = 0; i < n; i++)
        scanf("%d", &a[i]);
}

void outputArray(int *a, const int n) {
    for (int i = 0; i < n; i++)
        printf("%d ", a[i]);
    printf("\n");
}

void mergeArray(int *a, int *b, int *c, int n_a, int n_b, int *n_c) {
    for(size_t i = 0; i < n_a; i++) {
        c[i] = a[i];
        (*n_c)++;
    }

    for(size_t j = 0; j < n_b; j++) {
        int is_contain = 0;
        for(size_t k = 0; k < (*n_c); k++) {
            if(b[j] == c[k]) {
                is_contain = 1;
                break;
            }
        }
        if(is_contain == 0) {
            c[*n_c] = b[j];
            (*n_c)++;
        }
    }
}

int main() {
    SetConsoleOutputCP(CP_UTF8);

    int n_a;
    printf("Введите размер массива A: \n");
    scanf("%d", &n_a);

    int a[n_a];
    printf("Введите масива A: \n");
    inputArray(a, n_a);

    int n_b;
    printf("Введите размер массива B: \n");
    scanf("%d", &n_b);
```

```
int b[n_b];
    printf("Введите масива B: \n");
    inputArray(b, n_b);

    int n_c = 0;
    int c[n_a + n_b];

    mergeArray(a, b, c, n_a, n_b, &n_c);

    printf("Полученный массив C: \n");
    outputArray(c, n_c);
}
```

Вариант 2

Дано:

A – массив натуральных чисел, в котором нет одинаковых элементов;

B – массив натуральных чисел, в котором нет одинаковых элементов.

Получить массив C, содержащий все такие элементы, которые есть и в массиве A и в массиве B.

Решение:

```c
#include <stdio.h>
#include <windows.h>

void inputArray(int *a, const size_t n) {
    for (size_t i = 0; i < n; i++)
        scanf("%d", &a[i]);
}

void outputArray(int *a, const int n) {
    for (int i = 0; i < n; i++)
        printf("%d ", a[i]);
    printf("\n");
}

void doubleElementsArray(int *a, int *b, int *c, int n_a, int n_b, int *n_c)
{
    for(size_t i = 0; i < n_a; i++) {
        for(size_t j = 0; j < n_b; j++) {
            if(a[i] == b[j]) {
                c[(*n_c)++] = a[i];
            }
        }
    }
}

int main() {
    SetConsoleOutputCP(CP_UTF8);

    int n_a;
    printf("Введите размер массива A: \n");
    scanf("%d", &n_a);

    int a[n_a];
    printf("Введите масива A: \n");
    inputArray(a, n_a);

    int n_b;
    printf("Введите размер массива B: \n");
    scanf("%d", &n_b);

    int b[n_b];
    printf("Введите масива B: \n");
    inputArray(b, n_b);

    int n_c = 0;
    int c[n_a + n_b];

    doubleElementsArray(a, b, c, n_a, n_b, &n_c);

    printf("Полученный массив C: \n");
    outputArray(c, n_c);
}
```

Вариант 3

Дано:

A – массив натуральных чисел, в котором нет одинаковых элементов;

B – массив натуральных чисел, в котором нет одинаковых элементов.

Получить массив C, содержащий все элементы массива A, которых нет в B.

Решение:

```c
#include <stdio.h>
#include <windows.h>

void inputArray(int *a, const size_t n) {
    for (size_t i = 0; i < n; i++)
        scanf("%d", &a[i]);
}

void outputArray(int *a, const int n) {
    for (int i = 0; i < n; i++)
        printf("%d ", a[i]);
    printf("\n");
}

void getElementsFromBothArrays(int *a, int *b, int *c, int n_a, int n_b, int
*n_c) {
    for(size_t i = 0; i < n_a; i++) {
        int is_contain = 0;
        for(size_t j = 0; j < n_b; j++) {
            if(a[i] == b[j]) {
                is_contain = 1;
                break;
            }
        }
        if(is_contain == 0) {
            c[*n_c] = a[i];
            (*n_c)++;
        }
    }
}

int main() {
    SetConsoleOutputCP(CP_UTF8);

    int n_a;
    printf("Введите размер массива A: \n");
    scanf("%d", &n_a);

    int a[n_a];
    printf("Введите масива A: \n");
    inputArray(a, n_a);

    int n_b;
    printf("Введите размер массива B: \n");
    scanf("%d", &n_b);

    int b[n_b];
    printf("Введите масива B: \n");
    inputArray(b, n_b);

    int n_c = 0;
    int c[n_a + n_b];
```

```
    getElementsFromBothArrays(a, b, c, n_a, n_b, &n_c);

    printf("Полученный массив C: \n");
    outputArray(c, n_c);
}
```

Вариант 4

Дано:

A – массив натуральных чисел, в котором нет одинаковых элементов;

B – массив натуральных чисел, в котором нет одинаковых элементов.

Получить массив C, содержащий все элементы массива A, которых нет в B и все элементы массива B, которых нет в A.

Решение:

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

void inputArray(int *a, const size_t n) {
    for (size_t i = 0; i < n; i++)
        scanf("%d", &a[i]);
}

void outputArray(int *a, const int n) {
    for (int i = 0; i < n; i++)
        printf("%d ", a[i]);
    printf("\n");
}

void getUniqueElements(int *a, int *b, int *c, int n_a, int n_b, int *n_c) {
    for(size_t i = 0; i < n_a; i++) {
        int is_contain = 0;
        for(size_t j = 0; j < n_b; j++) {
            if(a[i] == b[j]) {
                is_contain = 1;
                break;
            }
        }
        if(is_contain == 0) {
            c[*n_c] = a[i];
            (*n_c)++;
        }
    }

    for(size_t i = 0; i < n_b; i++) {
        int is_contain = 0;
        for(size_t j = 0; j < n_a; j++) {
            if(b[i] == a[j]) {
                is_contain = 1;
                break;
            }
        }
        if(is_contain == 0) {
            c[*n_c] = b[i];
            (*n_c)++;
        }
```

```
    }
}

int main() {
    SetConsoleOutputCP(CP_UTF8);

    int n_a;
    printf("Введите размер массива A: \n");
    scanf("%d", &n_a);

    int a[n_a];
    printf("Введите масива A: \n");
    inputArray(a, n_a);

    int n_b;
    printf("Введите размер массива B: \n");
    scanf("%d", &n_b);

    int b[n_b];
    printf("Введите масива B: \n");
    inputArray(b, n_b);

    int n_c = 0;
    int c[n_a + n_b];

    getUniqueElements(a, b, c, n_a, n_b, &n_c);

    printf("Полученный массив C: \n");
    outputArray(c, n_c);
}
```

Вариант 5

Дано:

A – массив натуральных чисел, в котором нет одинаковых элементов;

B – массив натуральных чисел, в котором нет одинаковых элементов.

Определить, верно ли, что массив B содержит каждый элемент массива A.

Решение:

```c
#include <stdio.h>
#include <windows.h>

void inputArray(int *a, const size_t n) {
    for (size_t i = 0; i < n; i++)
        scanf("%d", &a[i]);
}

void outputArray(int *a, const int n) {
    for (int i = 0; i < n; i++)
        printf("%d ", a[i]);
    printf("\n");
}

int isContainsEachElements(int const *a, int const *b, int n_a, int n_b) {
    for(size_t i = 0; i < n_a; i++) {
        int is_contain = 0;
        for(size_t j = 0; j < n_b; j++) {
            if(a[i] == b[j]) {
                is_contain = 1;
            }
        }
        if(is_contain == 0) {
            return 0;
        }
    }
    return 1;
}

int main() {
    SetConsoleOutputCP(CP_UTF8);

    int n_a;
    printf("Введите размер массива A: \n");
    scanf("%d", &n_a);

    int a[n_a];
    printf("Введите масива A: \n");
    inputArray(a, n_a);

    int n_b;
    printf("Введите размер массива B: \n");
    scanf("%d", &n_b);

    int b[n_b];
    printf("Введите масива B: \n");
    inputArray(b, n_b);

    printf("Верно ли, что массив B содержит каждый элемент массива A: %d\n",
isContainsEachElements(a, b, n_a, n_b));

}
```

Вариант 6

Дано:

A – массив натуральных чисел, в котором нет одинаковых элементов;

B – массив натуральных чисел, в котором нет одинаковых элементов.

Определить, верно ли, что массивы A и B состоят из одинаковых элементов.

Решение:

```c
#include <stdio.h>
#include <windows.h>

void inputArray(int *a, const size_t n) {
    for (size_t i = 0; i < n; i++)
        scanf("%d", &a[i]);
}

int isEqualArray(int *a, int *b, int n_a, int n_b) {
    if(n_a != n_b) {
        return 0;
    } else {
        for (int i = 0; i < n_a; i++) {
            int is_equal = 0;
            for(size_t j = 0; j < n_b; j++) {
                if (a[i] == b[j]) {
                    is_equal = 1;
                }
            }
            if(is_equal == 0) {
                return 0;
            }
        }
    }
    return 1;
}

int main() {
    SetConsoleOutputCP(CP_UTF8);

    int n_a;
    printf("Введите размер массива A: \n");
    scanf("%d", &n_a);

    int a[n_a];
    printf("Введите масива A: \n");
    inputArray(a, n_a);

    int n_b;
    printf("Введите размер массива B: \n");
    scanf("%d", &n_b);

    int b[n_b];
    printf("Введите масива B: \n");
    inputArray(b, n_b);

    printf("Верно ли, что массивы A и B состоят из одинаковых элементов: %d",
isEqualArray(a, b, n_a, n_b));

    return 0;
}
```

Вариант 7

Дано:

A – массив натуральных чисел, в котором нет одинаковых элементов;

B – массив натуральных чисел, в котором нет одинаковых элементов.

Определить, верно ли, что в массивах A и B нет общих элементов.

Решение:

```c
#include <stdio.h>
#include <windows.h>

int inputArray(int *a, int n) {
    for(size_t i = 0; i < n; i++) {
        scanf("%d", &a[i]);
    }
}

int noCommonElements(int const *a, int const *b, int max_n, int min_n) {
    for(size_t i = 0; i < min_n; i++) {
        for(size_t j = 0; j < max_n; j++) {
            if(a[i] == b[j]) {
                return 0;
            }
        }
    }
    return 1;
}


int main() {
    SetConsoleOutputCP(CP_UTF8);

    int n_a;
    printf("Введите размер массива A: \n");
    scanf("%d", &n_a);

    int a[n_a];
    printf("Введите масива A: \n");
    inputArray(a, n_a);

    int n_b;
    printf("Введите размер массива B: \n");
    scanf("%d", &n_b);

    int b[n_b];
    printf("Введите масива B: \n");
    inputArray(b, n_b);

    if(n_a > n_b) {
        printf("Верно ли, что в массивах A и B нет общих элементов: %d",
noCommonElements(b, a, n_a, n_b));
    } else {
        printf("Верно ли, что в массивах A и B нет общих элементов: %d",
noCommonElements(a, b, n_b, n_a));
    }

}
```

Вариант 8

Даны массивы натуральных чисел A и B, упорядоченные по возрастанию.

Получить упорядоченный по возрастанию массив C, содержащий все элементы массивов A и B.

Решение:

```c
#include <stdio.h>
#include <windows.h>

void inputArray(int *a, const size_t n) {
    for (size_t i = 0; i < n; i++)
        scanf("%d", &a[i]);
}

void outputArray(int *a, const int n) {
    for (int i = 0; i < n; i++)
        printf("%d ", a[i]);
    printf("\n");
}

void containsElements(const int *a, const int *b, int *c, int n_a, int n_b){
    size_t count_a = 0;
    size_t count_b = 0;

    for(size_t i = 0; i < n_b + n_a; i++){
        if(a[count_a] >= b[count_b]) {
            c[i] = b[count_b];
            count_b++;
        }
        else {
            c[i] = a[count_a];
            count_a++;
        }
    }
}

int main() {
    SetConsoleOutputCP(CP_UTF8);

    int n_a;
    printf("Введите размер массива A: \n");
    scanf("%d", &n_a);

    int a[n_a];
    printf("Введите масива A: \n");
    inputArray(a, n_a);

    int n_b;
    printf("Введите размер массива B: \n");
    scanf("%d", &n_b);

    int b[n_b];
    printf("Введите масива B: \n");
    inputArray(b, n_b);

    int size_c = n_a + n_b;
    int c[size_c];

    containsElements(a, b, c, n_a, n_b);

    printf("Полученный массив C: \n");
```

```
    outputArray(c, size_c);
}
```

Вариант 9

Даны массивы натуральных чисел A и B, упорядоченные по возрастанию.

Получить упорядоченный по возрастанию массив C, содержащий все такие элементы, которые есть и в массиве A и в массиве B.

Решение:

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

void inputArray(int *a, const size_t n) {
    for (size_t i = 0; i < n; i++)
        scanf("%d", &a[i]);
}

void outputArray(int *a, const int n) {
    for (int i = 0; i < n; i++)
        printf("%d ", a[i]);
    printf("\n");
}

void containsDuplicateElements(const int *a, const int *b, int *c, const
size_t n_a, const size_t n_b, int *size_c){
    int index_a = 0;
    int index_b = 0;

    for(size_t i = 0; i < n_b && i < n_a; i++){
        if(a[index_a] < b[index_b]) {
            index_a++;
        } else if(a[index_a] > b[index_b]) {
            index_b++;
        } else {
            c[(*size_c)++] = a[index_a++];
            index_b++;
        }
    }
}

int main() {
    SetConsoleOutputCP(CP_UTF8);

    int n_a;
    printf("Введите размер массива A: \n");
    scanf("%d", &n_a);

    int a[n_a];
    printf("Введите масива A: \n");
    inputArray(a, n_a);

    int n_b;
    printf("Введите размер массива B: \n");
    scanf("%d", &n_b);

    int b[n_b];
    printf("Введите масива B: \n");
    inputArray(b, n_b);

    int size_c = 0;
```

```
    int c[n_a + n_b];

    containsDuplicateElements(a, b, c, n_a, n_b, &size_c);

    printf("Полученный массив C: \n");
    outputArray(c, size_c);

    return 0;
}
```

Вариант 10

Даны массивы натуральных чисел A и B, упорядоченные по возрастанию.

Получить упорядоченный по возрастанию массив C, содержащий все элементы массива A, которых нет в B.

Решение:

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

void inputArray(int *a, const size_t n) {
    for (size_t i = 0; i < n; i++)
        scanf("%d", &a[i]);
}

void outputArray(int *a, const int n) {
    for (int i = 0; i < n; i++)
        printf("%d ", a[i]);
    printf("\n");
}

void containsUniqueA(const int *a, const int *b, int *c, int n_a, int n_b,
int *n_c){
    int index_a = 0;
    int index_b = 0;

    for(size_t i = 0; i < n_b && i < n_a; i++){
        if(a[index_a] > b[index_b]) {
            index_b++;
        } else if(b[index_b] == a[index_a]) {
            index_a++;
            index_b++;
        } else {
            c[(*n_c)++] = a[index_a++];
        }
    }
}

int main() {
    SetConsoleOutputCP(CP_UTF8);

    int n_a;
    printf("Введите размер массива A: \n");
    scanf("%d", &n_a);

    int a[n_a];
    printf("Введите масива A: \n");
    inputArray(a, n_a);

    int n_b;
    printf("Введите размер массива B: \n");
```

```
    scanf("%d", &n_b);

    int b[n_b];
    printf("Введите масива B: \n");
    inputArray(b, n_b);

    int n_c = 0;
    int c[n_a + n_b];

    containsUniqueA(a, b, c, n_a, n_b, &n_c);

    printf("Полученный массив C: \n");
    outputArray(c, n_c);

    return 0;
}
```

Вариант 11

Даны массивы натуральных чисел A и B, упорядоченные по возрастанию.

Получить упорядоченный по возрастанию массив C, содержащий все элементы массива A, которых нет в B и все элементы массива B, которых нет в A.

Решение:

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

void inputArray(int *a, const size_t n) {
    for (size_t i = 0; i < n; i++)
        scanf("%d", &a[i]);
}

void outputArray(int *a, const int n) {
    for (int i = 0; i < n; i++)
        printf("%d ", a[i]);
    printf("\n");
}

void containsUnique(const int *a, const int *b, int *c, int n_a, int n_b, int
*n_c){
    int index_a = 0;
    int index_b = 0;

    for(size_t i = 0; i < n_b && i < n_a; i++){
        if(a[index_a] > b[index_b]) {
            index_b++;
        } else if(b[index_b] == a[index_a]) {
            index_a++;
            index_b++;
        } else {
            c[(*n_c)++] = a[index_a++];
        }
    }

    for(size_t i = 0; i < n_b && i < n_a; i++){
        if(b[index_b] > a[index_a]) {
            index_a++;
        } else if(b[index_b] == a[index_a]) {
            index_a++;
            index_b++;
        } else {
```

```
            c[(*n_c)++] = b[index_b++];
        }
    }
}

int main() {
    SetConsoleOutputCP(CP_UTF8);

    int n_a;
    printf("Введите размер массива A: \n");
    scanf("%d", &n_a);

    int a[n_a];
    printf("Введите масива A: \n");
    inputArray(a, n_a);

    int n_b;
    printf("Введите размер массива B: \n");
    scanf("%d", &n_b);

    int b[n_b];
    printf("Введите масива B: \n");
    inputArray(b, n_b);

    int n_c = 0;
    int c[n_a + n_b];

    containsUnique(a, b, c, n_a, n_b, &n_c);

    printf("Полученный массив C: \n");
    outputArray(c, n_c);

    return 0;
}
```

Вариант 12

Даны массивы натуральных чисел A и B, упорядоченные по возрастанию.

Определить, верно ли, что массив B содержит каждый элемент массива A.

Решение:

```c
#include <stdio.h>
#include <windows.h>

void inputArray(int *a, const size_t n) {
    for (size_t i = 0; i < n; i++)
        scanf("%d", &a[i]);
}

void outputArray(int *a, const int n) {
    for (int i = 0; i < n; i++)
        printf("%d ", a[i]);
    printf("\n");
}

int containsEachElements(const int *a, const int *b, int n_a, int n_b){
    int index_a = 0;
    int index_b = 0;

    for(size_t i = 0; i < n_b + n_a; i++){
        if(a[index_a] < b[index_b]) {
            return 0;
        } else if(a[index_a] == b[index_b]) {
            index_a++;
        }
        index_b++;
    }

    return index_a == n_a;
}

int main() {
    SetConsoleOutputCP(CP_UTF8);

    int n_a;
    printf("Введите размер массива A: \n");
    scanf("%d", &n_a);

    int a[n_a];
    printf("Введите масива A: \n");
    inputArray(a, n_a);

    int n_b;
    printf("Введите размер массива B: \n");
    scanf("%d", &n_b);

    int b[n_b];
    printf("Введите масива B: \n");
    inputArray(b, n_b);

    printf("Верно ли, что массив B содержит каждый элемент массива A: %d",
containsEachElements(a, b, n_a, n_b));

    return 0;
}
```