Семинар #3: Массивы. Классные задачи.

Объявление, инициализация и присваивание:

```
#include <stdio.h>
int main()
{
        // Массив из 10 чисел типа int. Без инициализации ( значения могут быть какими угодно )
        int a[10];
        // Упрощенно можно считать, что мы создали сразу 10 переменных типа int под именами
        // a[0], a[1], a[2], ... a[9]
        // Считаем числа в массив с помощью scanf
        for (int i = 0; i < 10; i++)
                scanf("%d", &a[i]);
        // Если вы знаете, что будет хранится в массиве, то можно сразу его инициализировать
        int b[10] = {6, 43, 11, 57, 91} // оставшиеся( 5 чисел будут установлены нулями)
        int c[] = {5, 4, 7}; // Размер массива будет автоматически установлен как 3
        c = {8, 7, 9};//Это ошибка! Фигурными скобачками можно пользоваться только при создании
}
```

1. **Инициализация:** Объявить массив под названием `numbers` с 100 элементами типа `int` и инициализировать его следующими значениями: 4, 8, 15, 16, 23, 42, все остальные элементы – нули.
2. **Печать:** Напечатать содержимое массива numbers на экран (все 100 чисел). Числа должны быть напечатаны в одну строку через пробел.
3. **Изменить** значение 70-го элемента с нуля на 123. Проверить изменения, напечатав массив на экран.
4. **Без инициализации:** Объявите массив из 100 элементов без инициализации и напечатайте его.
5. **Нулевая инициализации:** Объявите массив из 100 элементов, проинициализировав их все нулями и напечатайте его.
6. **Присваивание:** Задать массив `numbers` квадратами номера элемента и распечатать его.

Основные алгоритмы:

Пример программы, которая считывает массив и печатает максимальный элемент.

```
#include <stdio.h>
#define MAXSIZE 1000
int main()
{
        int n;
        scanf("%d", &n);
        int array[MAXSIZE];
        for (int i = 0; i < n; i++)
                scanf("%d", &array[i]);
        int max = array[0];
        for (int i = 1; i < n; ++i)
                if (array[i] > max)
                        max = array[i];
        printf("Maximum element = %d\n", max);
}
```

- `#define MAXSIZE 1000` - Задаём константу-макрос MAXSIZE = 1000
- Дальше мы считываем `n` - число элементов массива. Затем, в цикле, считываем `n` элементов. Мы будем использовать первые `n` элементов массива. Остальные `(MAXSIZE - n)` элементы использовать не будем. Понятно, что `n` должно быть меньше либо равно `MAXSIZE`.
- Алгоритм вычисления максимума – проходим по массиву, храня максимальный элемент из пройденых элементов массива.

Задачи:

1. **Сумма:** Написать программу, которая будет считывать массив и печатать сумму всех элементов.
2. **Каждый второй:** Написать программу, которая будет печатать каждый второй элемент массива.
3. **Синусы:** На вход подаётся число `n` и, затем, n углов(в радианах). Напечатать синусы этих углов. Использовать тип `float`. (Не забудьте `math.h` и опцию `-lm` для `gcc`.)
4. **Перестановка:** Напишите программу, которая будет перестанавливать 2 элемента. На вход идут количество элементов, сами элементы и 2 индекса (номера) элементов, которые нужно переставить. Например при входе:

   ```
   4
   2 4 8 16
   0 2
   ```

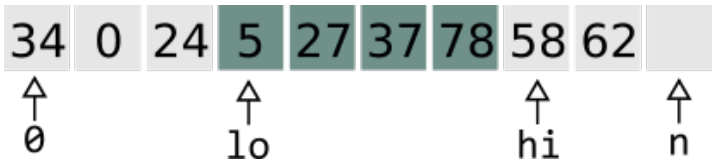
   Должно напечатать `8 4 2 16`.

Передача массивов в функцию

```c
#include <stdio.h>
// Массивы в функцию всегда передаются через указатель (int arr[] и int* arr одно и то же)
void print_array(int* arr, int n)
{
        for (int i = 0; i < n; i++)
                printf("%d ", arr[i]);
        printf("\n");
}
// Следовательно, при изменении массива в функции, он меняется и вне функции
void square_array(int arr[], int n)
{
        for (int i = 0; i < n; i++)
                arr[i] = arr[i] * arr[i];
}
int main()
{
        // Часто, мы не знаем размер массива заранее, поэтому берём с запасом
        // (1000 элементов) и работаем только с n первыми элементами
        int a[1000];
        int n;
        scanf("%d", &n);
        for (int i = 0; i < n; i++)
                scanf("%d", &a[i]);
        print_array(a, n);
        square_array(a, n);
        print_array(a, n);
}
```

1. **Сумма массива:** Написать функцию `int sum(int* arr, int n)`, которая будет возвращать сумму массива целых чисел. Входные параметры такие же как и предыдущей задаче. Функция не должна ничего печатать и считывать. Использовать эту функцию в функции main() чтобы найти сумму чисел массива.

2. **Сумма подмассива:** Написать функцию `int sum(int* arr, int lo, int hi)`, которая будет возвращать сумму подмассива `[lo:hi]`. Функция не должна ничего печатать и считывать. Использовать эту функцию в функции main().



3. **Умножение массива:** Написать функцию `void multiply(float* arr, int n, float k)`, которая будет умножать все элементы массива на число `k`. Протестируйте эту функцию в `main`.

4. **Индекс минимального элемента:** Написать функцию `int min_index(int* arr, int n)`, которая будет возвращать *индекс* наименьшего числа в массиве. Входные параметры такие же как и предыдущей задаче. Функция не должна ничего печатать и считывать. Если в массиве есть несколько минимальных элементов, то функция должна вернуть индекс первого из них.

Сортировка выбором

1. **Сортировка выбором:** Написать функцию `void selection_sort(int* arr, int n)`, которая будет сортировать массив методом выбора.
   Алгоритм сортировки методом выбора:

   - Идём циклом с индексом `i` от `0` до `n`.
     - Находим индекс минимального элемента от `i` до `n`.
     - Переставляем местами элемент с индексом `i` и этот минимальный элемент

   Протестируйте работу функции на массиве из хотя бы 15-ти элементов.

```c
#include <stdio.h>
#define MAXSIZE 100
// Тут Вам нужно написать функции print_array и selection_sort:
...
int main()
{
        int a[MAXSIZE] = {86, 12, 44, -36, -32, 2, -10, -3, 39, 60, 79, 97, -17, -29, 93};
        print_array(a, 0, 15);
        selection_sort(a, 15);
        print_array(a, 0, 15);
}
```

2. **Сортировка выбором по убыванию:** Написать функцию `void selection_sort_descend(int* arr, int n)`, которая будет сортировать массив методом выбора по убыванию.

3. **Рекурсивная сортировка выбором:** Написать функцию `void selection_sort_rec(int* arr, int lo, int hi)`, которая будет сортировать часть массива от индекса `lo` до `hi` (не включая `hi`) методом выбора. Используйте рекурсию. Алгоритм сортировки методом выбора для сортировки подмассива `[lo:hi]`:

   - Если `hi - lo <= 1` (один элемент), то завершаем функцию (просто `return`).
   - Находим индекс минимального элемента в подмассиве `[lo:hi]`
   - Переставляем местами элемент с индексом `lo` и минимальный элемент
   - Повторяем то же самое для подмассива `[lo+1:hi]`

Двумерные массивы

```c
#include <stdio.h>
#define MAX 100
void print_array(int arr[MAX][MAX], int n, int m)
{
        for (int i = 0; i < n; i++) {
                for (int j = 0; j < m; j++)
                        printf("%d ", arr[i][j]);
                printf("\n");
        }
}
int main()
{
        // Создаём массивы с запасом
        int a[MAX][MAX] = {{7, 7, 2}, {1, 8, 3}, {2, 1, 6}};
        int b[MAX][MAX] = {{5, 2, 9}, {-4, 2, 11}, {7, 1, -5}};
        // Печатаем только 9 элементов
        print_array(a, 3, 3);
}
```

1. **Умножение на число:** Написать `void multiply_num(int arr[MAX][MAX], int n, int m, int x)` которая будет умножать двумерный массив на число.
2. **Сложение:** Написать `void sum(int A[MAX][MAX], int B[MAX][MAX], int C[MAX][MAX], int n, int m)` которая будет складывать матрицы и записывать их в `C`.
3. **Умножение:** Написать `void mult(int A[MAX][MAX], int B[MAX][MAX], int C[MAX][MAX],int n, int m)` которая будет перемножать матрицы `A` и `B` (строка на столбец) и записывать их в массив `C`.

Считывание из файла

Пример программы, которая считывает массив из файла и печатает его содержимое на экран (нужно создать файл `input.txt` с входными данными в вашей рабочей папке):

```c
#include <stdio.h>
#define MAX 100
int main()
{
    FILE* file = fopen("input.txt", "r");  // "r" -- read   "w" -- write
    int n;
    fscanf(file, "%d", &n); // fscanf(file, < то же самое, что и в scanf >
    int array[MAX];
    for (int i = 0; i < n; i++)
        fscanf(file, "%d", &array[i]);

    for (int i = 0; i < n; i++)
        printf("%d ", array[i]);
    fclose(file);
}
```

1. **Сортировка из файла:** Написать программу, которая будет считывать массив чисел из файла `numbers.txt` и записывать в файл `sorted.txt` отсортированные числа.
2. **Умножение матриц из файла:** Написать программу, которая будет считывать матрицы из файлов `mat1.txt` и `mat2.txt`, перемножать их и записывать в файл `mult.txt`.