Дополнительные задачи по программированию

Легенда

- 3 – простые задачи
- 4 – задачи средней сложности
- 5 – более сложные задачи

Задачи

- 3 Написать программу, которая выводит строку «HELLO WORLD!» на консоль.
- На вход подаются два целых числа
  - если первое больше второго, вывести их сумму
  - если они равны, вывести 0
  - если второе больше первого, вывести их произведение.
- 3 Найти и вывести максимальное число из трех целых чисел (числа вводятся с консоли).
- 3 Заполнить значениями, введенными с клавиатуры, одномерный массив целых чисел длиной 15 и вывести эти значения на консоль.
- 3 Заполнить значениями, введенными с клавиатуры, одномерный массив целых чисел длиной 15 и вывести эти значения на консоль в обратном порядке.
- 3 Найти и вывести максимальное число из 15 целых чисел (числа вводятся с консоли).
- 3 Найти и вывести первое отрицательное число из 3 введенных целых чисел.
- 3 Найти и вывести первое отрицательное число из 15 введенных целых чисел.
- 3 Найти и вывести последнее отрицательное число из 15 введенных целых чисел.
- 3 Найти и вывести среднее арифметическое 3 введенных целых чисел.
- 3 Найти и вывести среднее арифметическое 15 введенных целых чисел.
- 3 Найти и вывести индекс первого символа пробела из 3 введенных символов (пробел вводится обязательно).
- 3 Найти и вывести индекс первого символа пробела из 15 введенных символов (пробел вводится обязательно).
- 3 Посчитать и вывести количество пробелов в 15 введенных символах.
- 3 Посчитать и вывести количество пробелов и восклицательных знаков в 15 введенных символах
- 3 Вывести индексы пробелов в 15 введенных символах, если пробелов в символах не было, вывести символ “-”.
- 3 Используйте оператор switch. На вход программе подается один из трех символов: -, +, * и два целых числа. Выведите результат операции для первого и второго числа. (Например, на вход поступило '-', 30, 10. Программа должна вывести 20.)
- 3 На вход программе подается сначала число n, а после - n целых чисел. Требуется определить, упорядоченны ли числа по неубыванию. Вывести «Yes» или «No». Числа для обработки сохранить в массив.
- 3 На вход программе подается строка, представляющая собой одно слово из латинских букв. Требуется определить, является ли слово палиндромом (одинаково читающееся в обоих направлениях (anna)). Вывести «Yes» или «No».
- 3 Используйте оператор switch. На вход программе подается целое число меньше 10.

Программа должна вывести слово «корова» в правильном падеже.
- 3 Заполнить двумерный массив нулями и вывести его на консоль.
  - размера 10×10
  - размера 5×10
  - размера 10×5

Задачи на массивы (в задачах следует полагать, что на вход программе сначала подается количество элементов N⇐100, а после - N чисел. Массив создавать статически.)

- 3 Найти разницу между максимальным и минимальным числом в этом массиве.
- 3 Найти сумму элементов массива, расположенных до минимального элемента
- 3 Найти сумму элементов массива, расположенных после последнего элемента, равного нулю
- 3 Найти сумму модулей элементов массива, расположенных после максимального о модулю элемента
- 3 Найти сумму модулей элементов массива, расположенных после первого элемента, равного нулю
- 3 Найти произведение элементов массива, расположенных между первым и вторых нулевыми элементами
- 3 Найти сумму элементов массива, расположенных между первым и последним отрицательными элементами
- 3 Найти произведение элементов массива, расположенных между максимальным по модулю и минимальным по модулю элементами

Строки:

- 3 Определить длину строки, введенной пользователем.
- 4 Найти разницу между числом строчных и прописных (заглавных) букв в строке.
- 4 Дана строка состоящая из букв и цифр (считается, что больше одной цифры подряд идти не может). Требуется посчитать сумму этих цифр.
- 4 Дана строка состоящая из букв и цифр Требуется сформировать и вывести новую строку, состоящую только из этих цифр, разделенных пробелами.
- 4 Поменять местами слова в строке, состоящей из двух слов.
- 4 Добавить в строку пробелы после знаков препинания, если они там отсутствуют.
- 4 Требуется в произвольной строке удалить последнее слово, т.е. все символы после последнего пробела в строке.
- 4 Найти слово, стоящее в тексте под определенным номером, и вывести его первую букву.
- 4 Подсчитать количество слов во введенной пользователем строке. Для упрощения задачи считать, что слова разделяются исключительно одним пробелом, а первый и последний символы строки не являются пробелами.
- 4 Во введенной строке удалить все символы между первым и вторым вопросительным знаком. Сами знаки в троке оставить
- 4 Дана строка. Определите процентное отношение строчных и прописных букв к общему числу символов в ней.
- 4 Написать функцию int mystrlen(char *); вычисляющую длину переданной ей строки

- 4 На вход программе подается число n, x, а после - n целых чисел, являющиеся коэффициентами многочлена степени n. Коэффициенты заданы по убыванию степеней. Треуется вычислить и вывести на экран значение многочлена в точке x.

5 Встреча с Римлянами

Попав в 500 год до нашей эры, вы столкнулись с необходимостью перевести число, записанное арабскими цифрами в число, записанное римскими цифрами

Напишите программу, которая переводит число (⇐3000) из арабской записи в римскую.

Примеры:

1994 → MCMXCIV

123 → CXXIII

5 Часы

Представьте обычные цифровые часы, время на которых отображается на 7-ми сегментных индикаторах с учетом ведущего нуля. Например, 9:30 на таких часах отображается так:

Числа отображатся следующим образом:

Требуется написать программу, на вход которой подается целое число n, обозначающее количество зажженных сегментов в отображении некого времени. Програма должна по этому числу востановить время, которое отображается на часах. Время должно быть корректным (подразумевается 24 формат представления времени). Если это невозможно, программа должна вывести «Impossible». Если может быть несколько вариантов корректного времени, вывести любой.

Примеры:

Вход: 23

Выход: 09:30

Вход: 28

Выход: Impossible

Вход: 2

Выход: Impossible

5 Могила

Территория некоторого кладбища имеет ровную прямоугольную форму. На нем (не выходя за его границы) расположена прямоугольная церковь. Требуется определить, хватит ли на кладбище места для прямоугольной могилы? могила может касаться границы кладбища и/или церкви, но не должна перекрывать их.

На вход программе подаются 10 чисел:

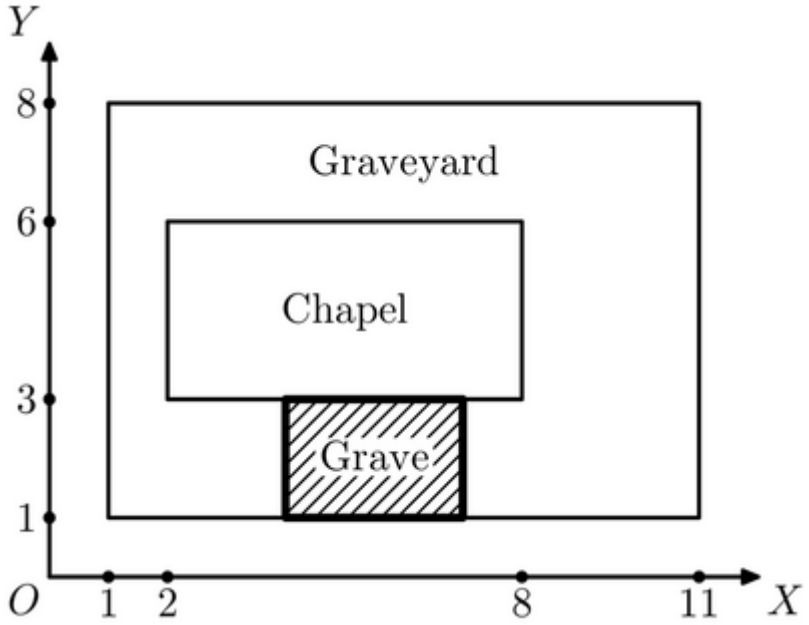
сначала **x1**, **y1**, **x2**, **y2** - координаты левого нижнего и правого верхнего углов гладбища

после **x1**, **y1**, **x2**, **y2** - координаты левого нижнего и правого верхнего углов церкви

а потом **w** и **h** - ширина и высота могилы. Сторона длины **w** должна располагаться вдоль оси OX, а сторона длины **h** - вдоль оси OY

Программа должна вывести «Yes», если место имеется и «No» в проиивном случае.

Пример:



Вход:

1 1 11 8

2 3 8 6

3 2

Выход:

Yes

Вход:

1 1 11 8

2 3 8 6

4 3

Выход:

No

5 Углы углы углы

Даны **n** попарно различных точек на плоскости с целыми координатами до 10^6 по абсолютной величине. Требуется найти среди этих точек две точки **A** и **B** такие, что угол **AOB** минимальный. **O** – центр координат.

5 Делим пространство

Дан набор прямых линий на плоскости (прямая может задаваться уравнением прямой или с помощью двух точек - на ваш выбор), которые могут пересекаться. Найти количество областей, на которые они разбивают плоскость.

4 Задачи на структуры В задачах подразумевается, что хранения связных данных (данных об одном логическом объекте) будет осуществляться с помощью типа struct.

- На вход программе подается число n, а после - n пар <Название журнала> <цена>. Следует полагать, что название не содержит пробельные символы. Требуется сохранить данные в массив и вывести информацию о журналах с ценой выше 100.
- На вход программе подается число n, а после - n пар <Название журнала> <цена>. Следует полагать, что название не содержит пробельные символы. Требуется создать и заполнить еще один массив теми журналами, стоимость которых меньше 30

Линейный односвязный список

Структуры: Способ представления данных инкапсулирующий в себя данные различных типов. Кратко можно почитать тут либо в какой-нибудь книге в духе «Руководство для начинающих (Шилдт)». Важно понимать, что данные-поля структуры в памяти представляются подряд, в порядке их объявления.

Идея: Идеологически - линейный односвязный список это структура данных, представляющая собой список узлов, каждый их которого хранит какие-то данные и указатель на следующий элемент. Это некая альтернатива массиву, но куда более гибкая: порядок элементов в списке не связан с их расположением в памяти. Это порождает, например, то, что такие операции как поиск и удаление не связаны со сдвигами остальных элементов, однако все операции со списком являются исключительно последовательными - прямой доступ к элементам списка невозможен. Кратко можно почитать тут. Либо в какой-нибудь книге по структурам данных.

Реализация Каждый узел линейного односвязного списка хранит в себе данные (которые подразумеваются под фразой «хранится в списке») и указатель на следующий элемент списка. Таким образом, узел списка может иметь вид:

node.h

```
struct node
{
  int data; //полезные данные
  struct node *next; //указатель на следующий элемент
};
```

Из этого следует то, что для всех операций со списком (в том числе и хранения его) достаточно знать только указатель на его первый элемент.

Функции: вам предлагается реализовать список и следующий набор функций (не обязательно все) для работы со списком (в списке хранятся целые числа). Не забывайте, что качество превыше количества, поэтому если сложно, то реализовать сколько сможете. Функции приведены в порядке возрастания сложности. В этом же порядке и реализовывать :

- Добавление элемента в конец списка (void addTail(struct node *root, int data);)
- Посчитать длину списка (int length(struct node *root);)
- Получить n-й элемент списка (int getN(struct node *root, int n);)
- Добавление элемента после n-го элемента списка (элементы нумеровать с нуля) (int addN(struct node *root, int n, int data); функция должна возвращать номер вставленного элемента и отрицательное число в случае ошибки)
- Добавление элемента в начало списка (void addHead(struct node *root, int data);) (сложнее, чем вам кажется)
- Подумать как можно реализовать удаление n-го элемента списка. (int remove(struct node *root, int n);)

Бинарное дерево поиска

Подробнее, например, тут

treeNode.h

```
struct node
{
  int data; //полезные данные
  struct node *left; //указатель на левого потомка
  struct node *right; //указатель на правого потомка
};
```

Функции:

Реализуйте следующие функции:

- Вставка элемента в дерево
- Поиск элемента в дереве
- Определение высоты дерева
- Вывод элементов дерева в порядке их возрастания (ЛКП обход дерева)
- Вывод элементов дерева с отображением структуры. Например:

```
((5)10(15))20((25)30(35))
```

- Удаление всех элементов дерева
- Удаление заданного элемента - рекомендую подумать и попробовать реализовать самостоятельно

Для удобства, рекомендуется использовать отдельную функцию для создания узла:

createNode.h

```
struct node* createNode(int n, struct node *left, struct node *right)
{
    struct node *cur = malloc(sizeof(struct node));
    cur->left = left;
    cur->right = right;
    cur->data = n;
    return cur;
}
```

Пример на обзор txt файлов во вложенных папках

dir-list.c

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

#include <sys/types.h>
#include <dirent.h>

void printFile(const char *path, const int lvl);

void dirTraveler(const char *startDir, const int lvl);

int main()
{
    dirTraveler(".",0);
    return 0;
}

void dirTraveler(const char *startDir, const int lvl)
{
    char path[1000];
    strcpy(path, startDir);

    DIR *dir=opendir(path);
    if(dir)
    {
    struct dirent *de = readdir(dir);
        while(de)
        {
            if(de->d_type == DT_REG && strstr(de->d_name,".txt"))
            {
                int path_len = strlen(path);
                strcat(path, "/");
                strcat(path, de->d_name);
                printFile(path, lvl);
```

```
                path[path_len] = '\0';
            }
            if(de->d_type == DT_DIR && 0!=strcmp(".", de->d_name) &&
0!=strcmp("..", de->d_name))
            {
                int path_len = strlen(path);
                strcat(path, "/");
                strcat(path, de->d_name);
                dirTraveler(path, lvl+1);
                path[path_len] = '\0';
            }
           de = readdir(dir);
        }
    }
    closedir(dir);
}

void printFile(const char *path, const int lvl)
{
    int i;
    char s[100];
    for(i=0;i<lvl;i++)
        printf("\t");
    printf("%s[\n", path);

    FILE *f = fopen(path, "r");
    if(f)
    {
        while(fgets(s,sizeof(s)/sizeof(char),f))
        {
            for(i=0;i<lvl+1;i++)
                printf("\t");
            printf("%s",s);
        }
        fclose(f);
    }

    for(i=0;i<lvl;i++)
        printf("\t");
    printf("]\n");
}
```

From:
https://se.moevm.info/ - **МОЭВМ Вики [se.moevm.info]**

Permanent link:
https://se.moevm.info/doku.php/courses:programming:extra_tasks

Last update: