Информатика

Кочанов Марк

22 сентября 2018 г.

МИФИ

Массивы

Объявление

```
1 int a[10];
2 char b[10], c[20];
```

- Нумерация элементов массива начинается с нуля.
- Длина массива должна быть явно указано (начиная со стандарта C99 допускается передавать в качестве длины массива переменную[1]).
- Массив не содержит внутри себя информации о своей длине.
- В языке C массивы по умолчанию не инициализируются.
- Копировать масивы нужно поэлементно.

[1] Функция alloca(), выделяющая память в стеке, приводит к проблемам при переполнении стека.

Инициализация

```
1  int a[5] = {0, 1, 2, 3, 4};
2  int b[10] = {0, 1, 2, 3, 4};
3  int c[10] = {0};
4  int d[10];
5
6  char str1[] = "string";
7  char str2[] = {'s', 't', 'r', 'i', 'n', 'g'};
8  char str3[] = {'s', 't', 'r', 'i', 'n', 'g', '\0'};
9  char str4[] = "qwerty\0qwerty";
10 char str5[20] = "string";
```

- Для хранения строк в языке C используются массивы с нулевым байтом для индикации конца строки (null-terminated string, C-строка).
- Нуль-символ используется всеми функциями для работы со строками из стандартной библиотеки для нахождения конца строки.

Работа со строками (string.h)

http://www.cplusplus.com/reference/cstring/

```
1  char str[30];
2  scanf("%s", str);
```

Введенная строка должна «умещаться» в выделенную память [1].

[1] https://stackoverflow.com/a/1621973

Многомерные массивы

```
1  int b1[2][2] = {{1, 2}, {3, 4}};
2  int b2[2][2] = {1, 2, 3, 4};
3
4  int b3[2][2] = {{1}, {3, 4}};
5  int b4[][2] = {1, 2, 3, 4}
```

- Для хранения многомерных массивов используется «row-major order».
- Весь массив располагается в памяти последовательно[2].

[2]Кроме случаев, когда происходит выравнивание данных.

Задачи

Simple linear regression (1/3)

Simple linear regression
Имеется набор пар чисел (x_i, y_i), по которым надо построить прямую с использованием метода наименьших квадратов. Тогда уравнение полученой прямой примет вид

$$f = \hat{\alpha} + \hat{\beta} x,$$
$$\hat{\alpha} = \bar{y} - \hat{\beta}\,\bar{x},$$
$$\hat{\beta} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2},$$

где $\bar{x}$ и $\bar{y}$ — среднее арифметическое соответствующих последовательностей.

Программа считывает из файла input.dat пары чисел (x_i, y_i) и выводит на экран значение коэффициентов $\hat{\alpha}$ и $\hat{\beta}$ через пробел. В первой строке файла содержится количество пар чисел. Далее на каждой строке располагается пара чисел, разделенных табуляцией или пробелом.

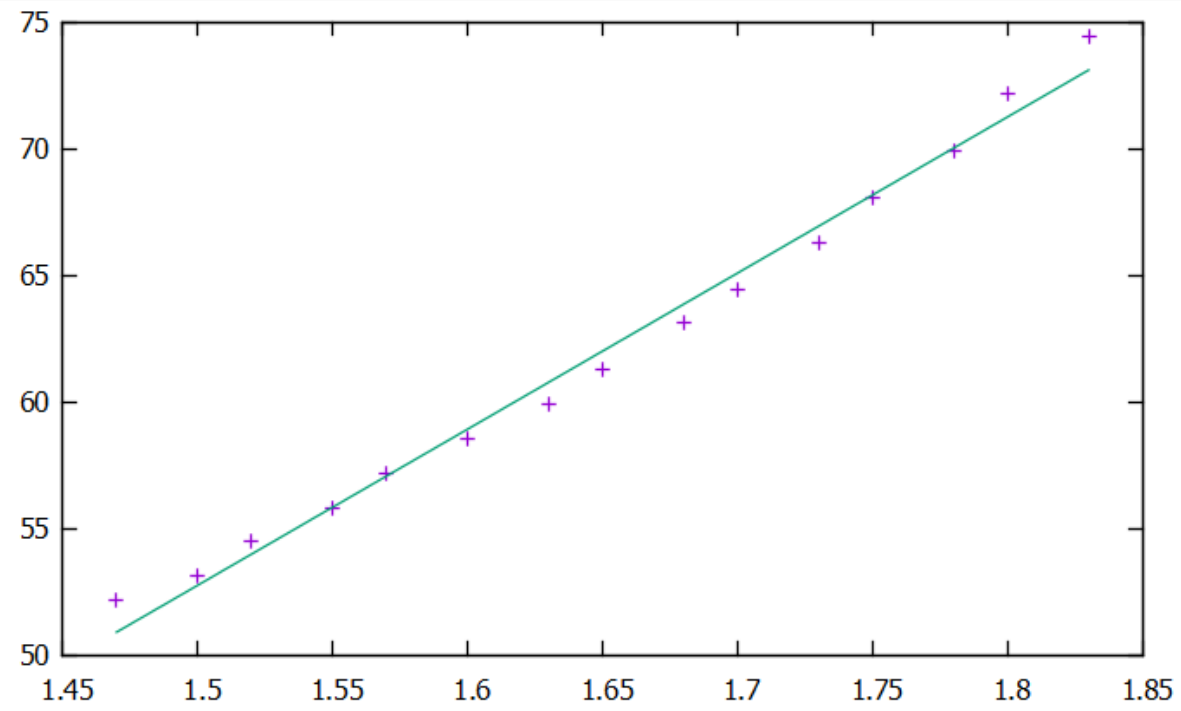
Simple linear regression (2/3)

Пример считывания чисел из файла.

```
1  FILE * file;
2  file = fopen("input.dat", "r");
3  if (file == NULL) {
4      exit (1);
5  }
6
7  int n;
8  float f1, f2;
9
10 fscanf(file, "%d", &n);
11 for (int i=0; i<n; i++) {
12     fscanf(file, "%f%f", &f1, &f2);
13 }
14
15 fclose(file);
```

Simple linear regression (3/3)

Скачать файл с примером данных



Двоичный (бинарный) поиск (1/2)

Программа при запуске спрашивает целое положительное число n. Далее необходимо сгенерировать отсортированный массив чисел (функция сортировки приведена на следующем слайде), после чего пользователь должен ввести число. В полученном массиве необходимо провести поиск искомого числа при помощи двоичного поиска. В случае наличия числа в массиве необходимо вывести его индекс, в противном случае вывести число -1.

Для создание неотсортированного массива использовать конгруэнтный метод (задача на 2 неделю), либо иной другой метод.

Двоичный (бинарный) поиск (2/2)

Пример сортировки массива при помощи функции qsort.

```c
#include <stdio.h>      /* printf */
#include <stdlib.h>     /* qsort */

int compare(const void *a, const void *b)
{
    return ( *(int*)a - *(int*)b );
}

int main()
{
    int values[] = {40, 10, 100, 90, 20, 25};

    qsort(values, 6, sizeof(int), compare);
    for (int n=0; n<6; n++)
        printf("%d ", values[n]);
    return 0;
}
```