Задачи за задължителна самоподготовка по Обектно-ориентирано програмиране

Шаблони и указатели към функции

email: kalin@fmi.uni-sofia.bg

28 май 2016 г.

1. Да се реализира шаблон на функция `void input ([подходящ тип] array, int n)`, която въвежда от клавиатурата стойностите на елементите на масива `array` от произволен тип `T` с големина `n`.

   Какви са допустимите типове `T` за този шаблон? Защо функцията е от тип `void`?

   Да се реализира и изпълни подходящ тест за функцията.

2. Да се реализира шаблон на фукнция `bool ordered ([подходящ тип] array, int n)`, която проверява дали елементите на масива `array` от произволен тип `T` с големина `n` образуват монотонно-растяща редица спрямо релацията `<`.

   Какви са допустимите типове `T` за този шаблон?

   Да се реализира и изпълни подходящ тест за функцията.

3. Да се реализира шаблон на фукнция `bool member ([подходящ тип] array, int n, [подходящ тип]x)`, която проверява дали `x` е елемент на масива `array` от произволен тип `T` с големина `n`.

Има ли в C++ тип `T`*, който не е съвместим с този шаблон?*

Да се реализира и изпълни подходящ тест за функцията.

4. Да се дефинира масив `functions` с 5 елемента от тип фунцкия *double* $\rightarrow$ *double*. Да се дефинират 5 произволни функции от този тип и адресите им да се се присвоят на елементите на масива.

   При въведено от клавиатурата число x : *double*, да се намери и отпечата индексът на тази функция в масива `functions`, чиято стойност е най-голяма в точката `x` спрямо стойностите на всички функции в масива. Ако има няколко такива функции, да се отпечата индекса на коя да е от тях.

5. Да се дефинира функция `double fmax([подходящ тип]f, [подходящ тип]g, double x)`, където `f` и `g` са две произволни функции от тип *double* $\rightarrow$ *double*, за които приемаме, че са дефинирани в `x`. Функцията да връща по-голямата измежду стойностите на `f` и `g` в точката x.

   Да се реализира и изпълни подходящ тест за функцията.

6. Да се дефинира фунцкия `double maxarray ([подходящ тип] array, int n, double x)`, където `array` е масив от финкции от тип *double* $\rightarrow$ *double* с големина `n`.

   Фунцкцията `maxarray` да връща най-голямата измежду стойностите на всички функции от масива в точката `x` като приемаме, че всички те са дефинирани в тази точка.

   Задачата да се реши със и без използването на функцията `fmax` от предишната задача.

   Да се реализира и изпълни подходящ тест за функцията.

7. Да се дефинира функция `void map (double array[], int n, [подходящ тип]f)`, където `array` е масив от стойности от тип *double* с големина `n`, а f е функция от тип *double* $\rightarrow$ *double*.

Функцията да заменя всяка стойност `array[i]` на масива `array` със стойността `f(array[i])`.

Например, ако е даден масивът `double a[] = {1,2,3}` и функцията `double inc (double x) {return x+1;}`, то след изпълнението на `map (a,3,inc)`, елементите на масива `a` ще имат стойности съответно 2, 3 и 4.

Да се реализира и изпълни подходящ тест за функцията.

8. Нека е дадена следната структура: `struct S {int a; int b; int c;};`. Да се дефинира функция `void sort([подходящ тип]array, int n, [подходящ тип]compare)`, където `array` е масив от `n` структури от тип `S`.

   Типът на функцията `compare` да се подбере така, че чрез нея да може да се реализира произволна наредба за типа `S`, т.е. функцията да може да сравнява "по големина" две структури от `S` по произволен критерий.

   Да се създаде и инициализира масив с 5 структури от тип `S`. Като се използва функцията `sort` да се сортира масива по веднъж по всеки от следните начини:

   (а) по полето `a`

   (б) по полето `b`

   (в) лексикографски по тройката (a, b, c)