Последна промяна: 06.12.2019, 21:05

СУ „Св. Климент Охридски“, ФМИ

Специалност „Софтуерно Инженерство“

Увод в програмирането, 2019-2020 г.

Задачи за домашно № 2

1. Известно е, че колоните в програмата Excel се именуват със символни низове (с дължина между 1 и 3 символа), които са съставени само от главни букви. Да се напише функция F()[1], която по даден низ *str*, намира поредния номер на колона (който съответства на str) в програмата Excel. Да се напише функция main(), в която потребителят въвежда символен низ. Да се изпълни F() върху въведеният низ и да се изведе резултата в конзолата.
   Забележка: Счита се че колоната с име “A” има номер 1, колоната с име “B” - номер 2, колоната с име “Z” - номер 26, колоната с име AB - номер 28 и т.н.

Примери:

Вход	**Изход**
D	4
AC	29
BA	53
XEO	16369

2. Да се напише функция F()[1], която приема като параметри символен низ str и две цели положителни числа M и L (M≤20, L<len, където len≤100 е дължината на str). Валидни символи за str са всички символи с ASCII кодове между 32 и 126 (включително). Функцията да прави опростено криптиране на низа, като всеки негов символ бъде преместен и заместен според следните условия (ако str съдържа невалидни символи, то той да остане непроменен):
   a. Преместването е на L позиции надясно в низа, като елементите се “превъртат” (ако новата позиция на даден символ е по-голяма от len, то броенето на позициите продължава от началото на str).
   b. Заместването се прави, като вместо оригиналният символ, на съответната позиция в низа се поставя нов елемент, чийто ASCII код е с M по-голям от ASCII кода на оригиналния символ.

   Да се напише функция main(), в която извиква функцията F() върху предварително въведен от потребителя (или локално дефиниран) масив и извежда в конзолата резултата от изпълнението на F().

[1] Сами определете подходящо име на функцията

Забележка: В повечето случаи въведеният от потребителя низ няма да съдържа невалидни символи, но функцията F() трябва да прави проверки и да не променя низа, ако той е невалиден.

Примери:

Вход	Резултат
str = “I am a string” M = 4, L = 3	mrkMeqe$wxv

str = “I\tam\na\rstring”
M = 4, L = 3
	I\tam\na\rstring
	
	// Низът съдържа
	// невалидни символи
	// и остава непроменен

3. Да се напише функция F()[2], която приема като входни параметри два непразни символни низа, с различна дължина и връща низ с дължина равна на дължината на по-големия от двата низа, който съдържа повторения на по-малкия низ. Да се напише функция main(), в която са декларирани локално два масива arr1 и arr2, която изпълнява F() върху тях и извежда резултата в конзолата.
Забележка: Когато дължината на по-дългия низ, не е достатъчна за пълно повторение на по-краткия, то вторият да се “отреже” до необходимия брой символи (виж третия пример: “we” се изрязва, като остава само първия символ).

Примери:

Вход	Резултат
arr1 = “qwerty” arr2 = “12”	“121212”
arr1 = “123” arr2 = “123 \t567”	“12312312”
arr1 = “we” arr2 = “we are the champions!”	“wewewewewewewewewewew”

4. Да се напише функция F()[2], която приема като параметър символен низ и намира броя на думите в него, които са с най-малка дължина, т.е. всички думи които имат дължина равна на най-късата дума в низа. Под дума се разбира всяка последователност от малки и/или главни букви от латинската азбука, както и символите тире "-" и долна черта "_", която е ограничена отляво и отдясно с интервал, табулация, началото или края на входящия низ. Да се напише функция

[2] Сами определете подходящо име на функцията

main(), в която е деклариран локално масива arr и да се изведе на екрана броят думи с най-малка дължина в него.
Забележка: Обърнете внимание, че между две валидни думи може да има повече от един разделителен символ (интервал, табулация, и т.н.).

Примери:

Вход	**Резултат**
`arr = “The best students are from FMI”`	`3 //The, are и FMI`
`arr = “free tree three c@r 4a6a perfect”`	`2 // 4, 6 и @ не са валидни // символи за дума, т.е. // с най-малка дължина // са free и tree`
`arr = “t_p”`	`1`
`arr = “a/3”`	`0 // В низа няма валидни // думи`
`arr = “The FMI \t rullzz_”`	`2 // The & FMI`

5. Да се напише функция F()[3], която по дадена квадратна матрица от естествени числа и нейният размер N (NxN, N<=10), проверява дали редовете в нея може да се пренаредят така, че по главния диагонал на матрицата да има само нарастващи числа (допускат се и две еднакви числа да са едно до друго по диагонала). Ако това е възможно, то матрицата да се пренареди, така че само с разместване на редовете в нея, по главния и диагонал числата да са нарастващи. Да се напише функция main(), в която е декларирана локално матрица от естествени числа и да се изведе на екрана резултата от изпълнението на F() върху нея.
Забележка: Ако съществуват повече от едно валидни пренареждания на матрицата, като валидно решение ще се възприема което и да е от тях.

Пример:

Вход	**Изход**
‖5 4 3 8 0 6‖ ‖9 4 5 1 0 8‖ ‖8 5 8 0 6 1‖ ‖0 9 6 8 4 2‖ ‖6 1 0 2 6 1‖ ‖2 1 8 4 2 6‖	‖0 9 6 8 4 2‖ ‖6 1 0 2 1 6‖ ‖5 4 3 8 0 6‖ ‖2 1 8 4 2 6‖ ‖8 5 8 0 6 1‖ ‖9 4 5 1 0 8‖

[3] Сами определете подходящо име на функцията

Последна промяна: 06.12.2019, 21:05

Последна промяна: 06.12.2019, 21:05

Пояснения:

1. Задача 1 носи 1 точка, задачи 2-4 – по 2 точки, а задача 5 – 3 точки.
2. В решенията на дадените задачи **не се допуска** използването на String и/или STL библиотеки, както и вградени функции за преобразуване на символ в число и обратно.
3. Всички задачи ще бъдат проверени автоматично за преписване. Файловете с голямо съвпадение ще бъдат проверени ръчно от преподавателите и при установено плагиатство ще бъдат анулирани.
4. Предадените от вас решения трябва да могат да се компилират успешно на Visual C++ или GCC
5. Всяка задача от домашното трябва да бъде решена в точно един, отделен файл. Името на файла трябва да бъде в следния формат:

fnXXXXX_d2_N_CC.cpp, където:

- XXXXX е вашият факултетен номер
- N е номерът на задачата
- CC указва кой компилатор сте използвали. Стойността му може да бъде “gcc” за GCC или “vc” за Visual C++.

6. Архивирайте всички файлове, които предавате в един архивен файл, компресиран в стандартен zip формат, със следното име:

UP_19-20_fnXXXXX_d2.zip, където XXXXX е вашият факултетен номер

7. Файловете с решенията, които предавате трябва да са оформени съгласно добрите практики за оформяне на кода, за които се говори по време на лекции и упражнения. Ще се отнемат точки за неинформативни имена на променливи, неизползване на подходящи константи и т.н.
8. Всички предадени програми трябва да се държат адекватно при некоректни входни данни от потребителя. (например въвеждане на поредица от символи, когато програмата очаква число).
9. Файловете с решенията може да съдържат само стандартните символи с кодове от 0-127 (не се разрешава използване на кирилица, например в стринговете или коментарите!).

10. Първото нещо във всеки от файловете, които предавате, трябва да бъде коментарен блок, който носи информация за съдържанието на файла. Този коментар трябва да изглежда точно така, както е показано по-долу, като в него попълните своите лични данни. За улеснение, просто копирайте дадения по-долу блок и попълнете в него нужната информация. Обърнете внимание, че на първия ред след наклонената черта има две звезди и че във файловете не може да се съдържат символи на кирилица.

```
/**
*
* Solution to homework assignment 2
* Introduction to programming course
* Faculty of Mathematics and Informatics of Sofia University
* Winter semester 2019/2020
*
* @author <вашето име>
* @idnumber <вашият факултетен номер>
* @task <номер на задача>
* @compiler <използван компилатор - GCC или VC>
*
*/
```

Например един попълнен блок за студент с име Иван Иванов, ф.н. 12345, който предава задача 2, компилирана с GCC, трябва да изглежда така:

```
/**
*
* Solution to homework assignment 2
* Introduction to programming course
* Faculty of Mathematics and Informatics of Sofia University
* Winter semester 2019/2020
*
* @author Ivan Ivanov
* @idnumber 12345
* @task 2
* @compiler GCC
*
*/
```

11. Предадени домашни, които не отговарят на условията от точки 2-10 ще бъдат оценени с 0 точки.