Последна промяна - 22 декември 2017 г., 15:40

СУ „Св. Климент Охридски“, ФМИ

Специалност „Софтуерно Инженерство“

Увод в прогамирането, 2017-2018 г.

Задачи за домашно № 2

1. Да се напише програма – игра, в която потребителят трябва да отгатне дума на английски език, като последователно се опитва да познае съставящите я букви. В началото, след стартиране на играта, думата се извежда на конзолата, като са изписани само първата и последната ѝ буква. На мястото на останалите букви на думата да се изведе символът ‘_’ (долна черта), но ако някои от вътрешните букви на думата съвпадат с първата или последната, те също трябва да се изпишат. Също така се извежда и броят опити за познаване на букви, които има играещият преди да загуби играта. След това играта протича по следния начин:
    a. Играещият прави предположение за това дали някоя буква се съдържа в думата, като въвежда от клавиатурата валиден символ (буква от латинската азбука, като не се прави разлика между главни и малки букви).
    b. В случай че въведената буква се съдържа в думата, то изведената в началото на играта дума се актуализира, като на съответните места в нея, долните черти се заместват с познатата буква. След това, ирачът прави ново предположение.
    c. В случай че въведената буква не е част от думата, то се намалява с единица броят опити за познаване на думата. Ако след актуализацията той е по-голям от нула, играещият прави ново предположение за буква, в противен случай – играта приключва.

    Играта продължава, докато играещият познае коя е думата или изчерпа правото си на грешки.

    Допълнителни условия:
    - За да бъде играта разнообразна, реализирайте своеобразна „база от думи“, като глобален масив с размер 30 английски думи. Думите в масива нека бъдат по ваш избор, като задължително трябва да фигурират следните две думи: “student” и “refrigerator”.
    - За достъп до този масив, използвайте една единствена функция, която случайно да избира някоя от думите в него. За целта може да използвате вградените в C++ библиотеки за генериране на случайни числа.
    - Направете различни нива на трудност в играта, според които играещият може сам да избира колко броя опити има, за да познае думата.

Пример:

Изход на конзолата	**Опит за познаване на буква**
st____t Attempts left: 8	a
Error! st____t Attempts left: 7	u
Success! stu___t Attempts left: 7	o
Error! stu___t Attempts left: 6	e
Success! stu_e_t Attempts left: 6	n
Success! stu_ent Attempts left: 6	d
Congratulations, you win! The word is 'student'.	

2. Напишете функция, която приема като аргумент изречение (до 100 символа) на латиница и след това извежда думите, които го съставят една под друга, заградени в рамка. (първия и последния ред на рамката са съставени от символът „тире“ ('-'), а вертикалните линии се изобразяват със символа '|'. При разчитането на изречението, за разделител между думите се приема един или повече интервали и/или табулация. Препинателните знаци да се изключат. За целта в програмата да се поддържа глобален масив с разпознаваемите препинателни знаци и обръщението към него да става само чрез специално написана за целта единствена функция.
Забележка: Препинателният знак се счита за разделител между две думи, независимо дали има интервал.

Пример:

Вход	**Изход**
She was an exceptional student.	---------- |She | |was | |an | |exceptional| |student | ----------

3. Да се напише функция, която по зададено цяло число, намиращо се в интервала [-2^{32}, 2^{32}], да намира цифрата, която се среща най-много пъти в десетичния запис на числото, както и броя на срещанията ѝ (сами преценете по какъв начин функцията ще връща описаните по-горе резултати). Ако числото е извън посочения интервал, да се връща резултат -1. Да се напише програма, която въвежда от клавиатурата цяло число, намиращо се в интервала [-2^{32}, 2^{32}], и след това да извежда на екрана цифрата (или цифрите, ако са повече от една), която се среща най-много пъти в десетичния запис на числото, както и броя на срещанията ѝ. За решаване на задачата не се допуска използването на вградените в езика функции за манипулация на символни низове.

   Примери:

Вход	**Изход**
436686522	6 -> 3
-444	4 -> 3
7000367400007	-1

4. Да се напише програма, която по въведен от клавиатурата низ (до 50 символа), намира и извежда на екрана всички негови под-низове, които са палиндроми, както и броят на срещанията им.

   Пример:

Вход	**Изход**
jagajoiojagaj	jagaj - 2 jagajoiojagaj - 1 aga – 2 agajoiojaga – 1 gajoiojag - 1 ajoioja - 1 joioj – 1 oio – 1
black sreen is not enough	ee - 1

5. Да се напише програма, която анимира падащи звезди в конзолата (приемаме черният ѝ цвят за нощно небе). Движението на звездите се анимира като вертикална линия изобразена чрез еднакви символи, намиращи се един под друг на няколко съседни реда (символът да може да се въвежда от клавиатурата). Всяка падаща звезда започва движението си от случайно избрана позиция в първия ред на конзолата, като дължината на опашката на всички звезди е 6 символа (т.е. 6 последователни реда). Движението на всяка зведа продължава до ред, който е случайно число между 6 и 25, след което тя и опашката ѝ изчезват.

   Допълнителни условия:

- Програмата завършва с натискане на клавишната комбинация "Ctrl+C".
- За решаването на задачата не се допуска използването на класове.
- Приема се, че екранът на конзолата има 25 реда и 80 символа на ред.
- За решаването на задачата може да се използват вградените възможности за изчистване на екрана на конзолата (system("cls")) и за изчакване (sleep()).
- Стартовите позиции на звездите не са уникални и може да се повтарят.
- За генериране на случайни числа може да използвате вградените в C++ библиотеки.
- Приема се, че в даден момент от време в небето има константен брой звезди който се въвежда от клавиатурата при стартирането на програмата (число между 3 и 80). В този смисъл при завършване на движението на дадена звезда, в „небето" (т.е. на конзолата) се появява следващата.

Пояснения:

1. Задачи 1 и 3 и 5 носят по 2 точки, задача 2 - 1 точка, а задача 4 – 3 точки.
2. За решаване на всички задачи от домашното не се допуска използването на string.
3. Всички задачи ще бъдат проверени автоматично за преписване. Файловете с голямо съвпадение ще бъдат проверени ръчно от лектора и при установено заимстване 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_17-18_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.
8. 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!).
9. 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 second homework task
* Introduction to programming course
* Faculty of Mathematics and Informatics of Sofia University
* Winter semester 2017/2018
*
* @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 second homework task
* Introduction to programming course
* Faculty of Mathematics and Informatics of Sofia University
* Winter semester 2017/2018
*
* @author Ivan Ivanov
* @idnumber 12345
* @task 2
* @compiler GCC
*
*/
```

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