Курсови проекти по Структури от данни

СУ-ФМИ, Специалност „Информационни системи“, 2 курс

Зимен семестър 2013/2014

Обща информация за проектите

Всеки един от дадените по-долу проекти получава като вход пълен или относителен път до една или повече директории (за всеки проект това е уточнено подробно). Пътят до директорията трябва да може да се подаде като аргумент от командния ред.

За **всички** проекти ще считаме, че:

- Входът на програмата се подава под формата на параметри от командния ред.
- Директориите могат да съдържат произволен брой поддиректории и файлове. Поддиректориите на свой ред могат да съдържат други поддиректории и файлове и т.н.
- За улеснение ще считаме, че в директориите няма твърди връзки (hard links), нито символни връзки (symbolic links или още soft links).

Към всеки от проектите могат да бъдат дадени допълнителни (бонус) точки, ако той бъде разширен и в него бъде включена по-сложна функционалност. Под всеки от проектите са дадени няколко възможности за получаване на бонус точки.

Проектите се оценяват по няколко критерия, всеки от които носи точки. Общата сума от точките надвишава необходимия брой за оценка отличен. Някои от основните критерии са:

- **Дали решението работи коректно**.
- **Дали решението използва подходящи алгоритми и структури от данни**. Brute force решения също се допускат, но носят минимален брой точки.
- **Дали решението отговаря на заданието на проекта**.
- **Оформление на решението**. Проверява се дали кодът е добре оформен, дали е спазена конвенция за именуване на променливите, дали е добре коментиран и т.н.
- **Дали проектът е документиран**. Проверява се дали кодът е добре коментиран и дали въз основа на коментарите автоматично се генерира техническа документация (напр. с Doxygen). Този критерий не е задължителен и носи бонус точки.
- **Дали решението е било добре тествано**. Проверява се какви тестове са били проведени за приложението (напр. unit test-ове). По този критерий се очаква по време на защитата да можете да посочите как сте тествали приложението и как сте проверили неговото поведение в различни ситуации.

Оценката на всеки от проектите се формира от онази негова част, която е била самостоятелно разработена от вас. Допустимо е да използвате код написан от някой друг (напр. готова библиотека или помощ от ваш приятел/колега), но това трябва да бъде ясно обявено при предаването и защитата на проекта, като посочите коя част от проекта сте разработили самостоятелно. Когато в даден проект се използва чужд код се оценяват и (1) способността ви за внедряване на този код във вашето решение (напр. в случаите, когато се използва външна библиотека) и (2) дали добре разбирате какво прави чуждият код.

1: Откриване на дублиращи се файлове

В рамките на този проект трябва да се разработи приложение, което сканира дадена директория и открива дублиращите се файлове. Под дублиращи се файлове се има предвид файлове, които имат идентично съдържание. Файловете могат да имат различни имена, атрибути и т.н., но ако съдържанието им съвпада, се счита, че те са дублиращи се.

Като вход приложението получава пълен или относителен път до една директория.

Приложението трябва да изведе информация за намерените дублиращи се файлове. Като минимум трябва да се изведе колко памет се губи от дублирането и кои са дублиращите се файлове. В изхода дублиращите се файлове трябва да се групират. Например ако приложението открие, че трите файла „test.txt”, “t.txt” и “something” се дублират, то трябва да изведе за тях само един запис на екрана, който може да бъде нещо подобно на следното:

```
Duplicate group 1
   (3,6MB, 1,2MB per file)
   C:\Temp\Test.txt
   C:\Temp\some\directories\here\t.txt
   C:\Temp\something
```

За допълнителни точки, приложението може да се разшири, като към него се добави следната функционалност:

1. Изходът да се извежда във вид на HTML
2. Програмата да дава предложения на потребителя за това кои файлове да изтрие (напр. да може да се избере да се посочат само по-новите файлове или само файлове в някоя поддиректория и т.н.)
3. Да се реализира почистване на дублиращите се файлове. Например с директно изтриване или преместване в някаква друга директория.

2: Синхронизация на директории

В рамките на този проект трябва да се напише приложение, което сканира две директории A и B и открива разликите между тях. След това приложението трябва да синхронизира двете директории.

За целите на синхронизирането, въвеждаме следната дефиниция за съответстващи си файлове: това са такива файлове, които имат еднакъв относителен път спрямо двете директории A и B. Например ако съдържанието на директориите е както следва:

• C:\Temp
  ○ MyFiles\
  ▪ A.txt
  ▪ B.txt
  ○ MyPhotos\
  ▪ File1.jpg
  ○ A.txt
	• C:\SomethingElse
	  ○ MyFiles\
	  ▪ A.txt
	  ▪ jhghgjgjg.txt
	  ○ MyPhotos\
	  ▪ A.txt
	  ○ Music.mp3

Съответстващи си файлове ще бъдат следните два:

```
C:\Temp\MyFiles\A.txt
C:\SomethingElse\MyFiles\A.txt
```

Това е така, защото относителни пътища до тях съвпадат (по-горе са подчертани). Въпреки, че в директориите има и други файлове с име A.txt, те се намират в други поддиректории.

За целите на синхронизирането програмата трябва да извършва следните операции:

1. Копира в B файловете и директориите, които се намират в A, но ги няма в B.
2. Копира в A файловете и директориите, които се намират в B, но ги няма в A.
3. За всяка двойка от съответстващи си файлове:
    a. Ако са идентични те не се променят
    b. Ако са различни по съдържание, тогава се избира един от файловете, който след това се презаписва върху другия.

Програмата трябва да получи като вход път към две директории. След това тя трябва да ги сканира и да изведе на екрана информация за действията, които ще извърши. Потребителят трябва да ги потвърди или отхвърли. Ако ги отхвърли нищо не се случва и програмата приключва своето изпълнение. Ако ги потвърди, програмата трябва да синхронизира двете директории.

За допълнителни точки, приложението може да се разшири, като към него се добави следната функционалност:

1. Да може да се избере режим на синхронизиране – или горе описания режим на сливане или режим „огледално копие“. В новия режим втората директория трябва да стане точно копие на първата. Тогава програмата работи така:
    a. Копира в B файловете и директориите, които се намират в A, но ги няма в B.
    b. Изтрива от B файловете и директориите, които се намират в B, но ги няма в A.
    c. За всяка двойка от съответстващи си файлове:
        i. Ако са идентични те не се променят
        ii. Ако са различни по съдържание, тогава се взима файлът от директория A и той се презаписва върху файла от директория B.

3: Архивиране на файлове

В рамките на този проект трябва да се разработи приложение, което сканира дадена директория и архивира нейното съдържание. Програмата трябва да създаде един архивен файл, който пази в себе си оригиналното съдържание на директорията. Съдържанието на файла трябва да се компресира с алгоритъма на Хъфман.

Приложението трябва да може да архивира и разархивира съдържанието на дадена директория (т.е. потребителят трябва да може или да укаже директория, която да се архивира или да укаже файл, който трябва да се разархивира, при което програмата трябва да възстанови от него оригиналното съдържание).

Като вход приложението получава един параметър, който указва каква операция да се изпълни – архивиране или разархивиране. В случай на архивиране, след него трябва да се

подадат път до директория, която да се обработи и името на архивния файл, който трябва да се създаде. При разархивиране трябва да се подаде път до съществуващ архив и път до директория, в която той да се разархивира. Можете сами да изберете формата на параметрите. Стартирането на програмата може да изглежда по начин подобен на тези по-долу:

```
Arc.exe /compress C:\Test C:\Temp\myfile.arc
Arc.exe /extract C:\Temp\myfile.arc C:\SomeOtherDirectory\
```

По време на изпълнението си, приложението трябва да извежда някаква информация, за да може потребителят да знае дали то работи или е „увиснало“. Например може да се извежда броят на обработените файлове, колко процента от работата е свършена и т.н.

За допълнителни точки, приложението може да се разшири, като към него се добави следната функционалност:

1. Да се реализира тестване на архив. Има се предвид операция, която получава като вход път към даден архив и проверява дали той може да бъде разархивиран, но без да се пише ново съдържание във файловата система.
2. Да се реализират и други алгоритми за компресия (напр. LZ77 или LZ78). При архивиране потребителят да може да избере каква компресия да се избере. При разархивиране приложението само трябва да открива кой алгоритъм е бил използван и няма нужда потребителят да указва това.