Септември испит по
Оперативни системи
16.09.2019

120

1. Да се напише PERL скрипта која како аргументи добива имиња на две датотеки. Во првата датотека е сместен текст кој треба да се обработи, додека пак, во втората датотека е сместен именик со контакти. Скриптата треба да ја измине првата датотека и да ги најде сите е-маил адреси (формат: kime@firma.com) и да креира команда спремна за испраќање на маил со конкретна порака. Командата се формира во следниов формат:

mail -s “poraka“ kime@firma.com -u Ime Prezime

За секоја емаил адреса што ќе се пронајде во влезната датотека, се креира ваква наредба, каде на местото од kime@firma.com се сместува конкретната најдена емаил адреса. Во втората датотека, чие што име се добива како влез на скриптата, е сместен именик со познати емаил адреси, така што во секој ред е сместена посебна емаил адреса, заедно со името и презимето на корисникот на емаил адресата (одделени со празно место). Скриптата треба да ја провери секоја пронајдена емаил адреса од првата датотека дали ја има во именикот креиран од втората датотека. Доколку пронајдената емаил адреса ја има во втората датотека, тогаш Ime и Prezime се заменуваат со името и презимето пронајдени во именикот за таа емаил адреса. Инаку, се игнорираат од наредбата . Параметарот “ poraka “ е текстуална порака чиј што текст е сместен во датотека со име „poraka.txt“. Доколку емаил адресата ја има во именикот, тогаш пред текстот од пораката се додава следниот текст: „Pocituvan Ime Prezime,“, каде име и презиме се земаат истите од претходно. Доколку емаил адресата ја нема во именикот, тогаш “poraka“ останува непроменета. Секоја ваква формирана наредба се печати на екран, притоа не се печати повеќе пати ако се пронајде истата емаил адреса повеќе пати.

Да се напише Shell скрипта која што ќе провери дали во тековниот директориум се наоѓаат датотеките „poraka.txt“ и „kontakti.vcf“. Доколку ги има, тогаш ги пребарува сите текстуални датотеки во тековниот директориум (со екстензија .txt, игнорирајќи ја датотеката „poraka.txt“) и секоја една датотека ја праќа на влез на повик на PERL скриптата заедно со „kontakti.vcf“. Секоја наредба што ќе се врати, SHELL скриптата треба да ја изврши.

2. Да се напише програма во C програмскиот јазик каде ќе се овозможи внесување на име на датотека како аргумент на програмата. Потоа, програмата треба да изврши мемориско мапирање на датотеката во адресниот простор. Датотеката е составена од N цели броеви. Главниот процес, треба да ја измине датотеката и да провери колку цели броеви има во (мемориски мапираната) датотека. Потоа, треба да креира 10 нитки, каде што секоја една од нитките ќе генерира случаен број од -10 до 2500, и ќе провери дали го има во датотеката. Доколку го има, бројот што е случајно генериран го сместува во низа, приота доколку некоја нитка повеќе пати го генерира истиот број, тогаш треба да се знае колку пати е секој еден број пронајден (генериран). Секоја нитка генерира 20 случајни броеви и потоа завршува. Откако ќе завршат сите нитки, главниот процес ги пренесува најдените броеви на дете процес, кое што тој го креира. Дете процесот е задолжено да отпечати на екран кои броеви биле пронајдени и колку пати во влезната датотека.

3. Да се напише драјвер за комуникација со уред. Уредот има меморија од 10KB, притоа се запишува во драјверот секогаш од лево на десно, притоа на следното запишување продолжува од каде што претходно застанал. Доколку стигне до крај со запишување, почнува од почеток на меморијата. Читањето се прави на истиот начин како и запишувањето, со тоа што, не може да прочита повеќе бајти од бројот на бајти последно запишани во драјверот. Пример: доколку последно се запишани 50 бајти во драјверот, а на читање се обиде да прочита повеќе од 50 бајти, драјверот ќе ги врати само првите 50 бајти.

Напомена: Да се користат следниве наредби за да се генерира случаен број од 0 до 100:

```
int i, randomNumber100;
get_random_bytes(&i, sizeof(i));
randomNumber100 = i % 100;
```

Јануарски испит по
Оперативни системи
Група 2
31.01.2019

120

1. Да се напише PERL скрипта која како аргумент добива име на датотека во која се запишани ценовник на производи. Во посебен ред од датотеката се наоѓа секој еден од производите, притоа во редот најпрво е баркодот на производот, потоа одделен со празно место е името на производот, а потоа исто така одделен со празно место е сместена цената на производот. Производите треба да се сместат во Хеш структура, каде што клуч ќе е баркодот, а како вредност ќе се чува името на производот. Треба да се креира и посебна Хеш структура каде како клуч ќе се земе повторно баркодот на производот, а како вредност ќе се чува цената на производот. Потоа, скриптата треба да дозволи да се внесуваат непознат број на команди од тастатура. Доколку од тастатура се внесе 1, тогаш низ ценовникот се пребарува по име на производ, што значи дека после тоа се внесува името на производот кој се пребарува а на екран се печати цената на производот (или дека производот не постои). Доколку се внесе 2, се листаат производите чија што цена им е помала од таа внесена од тастатура. Доколку се внесе 3, тогаш се печатат сите производи (секој производ во посебен ред) со името и цената на производот. Доколку се внесе вредност различна од 1, 2 или 3, тогаш се прекинува со командите и скриптата завршува.

   Да се напише Shell скрипта која што ги наоѓа сите ценовници во тековниот директориум т.е. ги бара сите датотеки кои што завршуваат на екстензија .vcf. Доколку има повеќе ценовници, тогаш на крајот во првиот ценовник од листата ценовници се додава содржината на другите ценовници и се повикува PERL скриптата со името на првиот ценовник.

2. Да се напише програма во C програмскиот јазик каде ќе се овозможи внесување на низа од цели броеви како аргументи на програмата. Потоа, програмата треба да дозволи внесување на непознат број на цели броеви за пребарување во низата. За секој внесен број, програмата креира нова нитка, преку која го пребарува тој број во низата и печати на екран колку пати се појавува тој број и колку има броеви во низата кои што се делители на тој број. Печатењето на екран не го прават нитките што го прават пребарувањето туку тие резултатот го препраќаат до друг процес преку мемориско мапирање. Внесувањето на броевите за пребарување прекинува кога ќе се внесе од тастатура нешто различно од цел број.

3. Да се напише драјвер за комуникација со уред. Уредот има меморија од 10KB, притоа се запишува во драјверот секогаш од лево на десно, притоа на следното запишување продолжува од каде што претходно застанал. Доколку стигне до крај со запишување, почнува од почеток на меморијата. Читањето се прави така што најпрво се запишува во драјверот или 1 или 2, па потоа се чита од драјверот. Доколку се запише 1, тогаш почнува да чита од почеток на меморијата, додека пак ако запише 2, тогаш чита од крајот на меморијата (последните K бајти). Доколку не се запише ништо а се обидеме да прочитаме од драјверот, тогаш одбира случајно од кој бајт да чита, притоа мора да одбере валиден бајт (т.е. да има доволно податоци десно од бајтот од каде треба да се чита.

Напомена: Да се користат следниве наредби за да се генерира случаен број од 0 до 100:

```
int i, randomNumber100;
get_random_bytes(&i, sizeof(i));
randomNumber100 = i % 100;
```