CS360 -- Lab 1

- Jian Huang
- CS360
- Url: http://www.cs.utk.edu/~huangj/cs360/360/labs/lab1/lab.html

---

Bu laboratuar sizin red-black tree ler, dllist ler ve field ler i anladığınızdan emin olmak içindir. Eğer henüz Dr. Plank'ın fdr kütüphanesiyle tanışmadıysanız(yada onu tazelemeye ihtiyacınız varsa), **Dr. Plank'ın libfdr ders notlarını okuyun.**

Bu laboratuar daki göreviniz famtree programını yazmaktır. Famtree insanlar ve onların birbirleriyle ilişkileri hakkında tanım alır. Tanımlar aşağıda açıklandığı gibi özel bir formatta olmalıdır. **fam1** deki örnek Bob, Nancy ve onların üç çocuğu olan Frank, Lester ve Diana nın oluşturduğu aile yi temsil eder.

Famtree böylesi bir tanımlama alır ve dosyadaki bütün kişiler üzerine tam bilgi çıktısı verir. Bu her kişi için onların cinsiyetleri(biliniyorsa), babaları, anneleri ve çocuklarından oluşur. Böylelikle **fam1output** **fam1** üzerine **famtree** nin geçerli çıktısını içerir.

Giriş dosyasının formatı şu şekilde olmalıdır. Satırlar ya boş olmalıdır yada şu ilk kelimelerden birini içermelidir:

- **PERSON**: Kişiyi belirtir. **PERSON** dan sonra kişinin ismi gelir. Bu herhangi bir sayı yada kelime olabilir. **Famtree** isimleri, aralarında bir boşluk bulunan kelimelerden oluştuğunu varsaymalıdır.
- **SEX**: Kişinin cinsiyetini belirler. Bu **M** yada **F** olabilir.
- **FATHER**: Kişinin babasını belirtir. **FATHER** dan sonra babanın ismi gelir. Bu herhangi bir numara yada kelime olabilir. Bu babanın erkek olduğunu ima eder. Bir kişinin sadece bir babası olabilir.
- **MOTHER**: Kişinin annesini belirtir. **MOTHER** dan sonra annenin ismi gelir. Bu herhangi bir numara yada kelime olabilir. Bu annenin kadın olduğunu ima eder. Bir kişinin sadece bir annesi olabilir.
- **FATHER_OF**: Bu kişinin erkek olduğunu belirtir, ve bu belirtilen kişi bir kişinin çocuklarından biridir. Bir kişinin çocuk sayısı herhangi bir numara olabilir.
- **MOTHER_OF**: Bu kişinin kadın olduğunu belirtir ve bu belirtilen kişi bir kişinin çocuklarından biridir.

Famtree diğer iki özelliğe sahiptir. Birincisi, kişileri yapısal bir sırada yazdırır.Bu hiç kimsenin ebeveynleri yazdırılmadan yazdırılamaması demektir. Eğer bu imkansızsa(örneğin **cyclefam** deki gibi) **famtree** bu gerçeği tanımlar.

famtree nin ikinci özelliği gereksizliğe izin verir. Ama yapabildiği kadar sonuca gider. Örneğin **redundant** birkaç gereksiz satırlara sahip. Örneğin 3. Satır gereksizdir çünkü Fred Joe nin babası olduğu için erkek olmalıdır. Bundan başka 7. Satır da gereksizdir çünkü 2. Satır zaten Fred in Joe nin babası olduğunu belirtir. **nonredundant** dosyası aynı aile için minimum belirtili dosyadır.

Çalışma örneği

Famtree nin çalışma örneği **~cs360/lab1/** dizinindedir. Bunu dizindeki bu giriş dosyalarıyla deneyin. Diğer giriş dosyaları:

- **fam2** daha kompleks bir ailedir.
- **fam3** tek ebeveynden 10 nesillik bir ailedir.
- **fam4** içinde hata olan bir dosyadır.
- **fam5** içinde hata olan başka bir dosyadır.

Çıktınızı tamamen **famtree** deki gibi yapmalısınız.

Not bölümlendirilmesi

Geçerli girişler için olan test(yani: **cyclefam** in geçersiz olduğunu gösteren test) hariç çalışan her şeyi yaptıysanız %90 not elde edebilirsiniz.

Doğru sırada çıktı almak bir diğer %20 nottur.Böylelikle, eğer doğru bilgilere göre kişileri bir sıraya göre basitçe yazdıran bir program yazdıysanız, %70 not alırsınız.

Yardım

Burada oldukça detaylı bilgi vereceğim. Doğrusu, temel olarak labarotuarın nasıl yapılacağını vereceğim. Bunun sonucunda programın doğru yapısını anlayacaksınız. Eğer bunu yardım almadan yapabileceğinizi düşünüyorsanız ve verilen cevabı istemiyorsanız, lütfen bunu okumayın ve eğlencenize bakın! Size; daha ilerisini okumadan önce, hiçbir yardım olmadan önce en azından bunu nasıl yapacağınızı düşünmenizi tavsiye ederim. Ancak, eğer biraz yardım istiyorsanız, okumaya devam edin.

Kişi için bir yapınız olacak. (benimki **Person**). Bu yapı şu alanlara sahip olacak:

- İsim
- Cinsiyet
- Baba
- Anne
- Çocukların Listesi
- Derinlik ve genişlik için bazı şeyler.

Programınız üç parçada çalışacak:

1. **Yapıların içine bilgiyi okumak.** Bütün kişileri içeren bir red-black tree niz olmalı. (benimki **people**). Her kişinin ismiyle anahtarlanmıştır ve val alanları kişilerin Person yapısıdır. Standart girişi okumak için IS kullanacaksınız.

Her defasında okuduğunuz satır bir isme sahiptir(Örneğin **PERSON**, **FATHER**, **MOTHER**, **FATHER_OF** ve **MOTHER_OF**). Eğer insan ağacındaki kişi bu isimdeyse test ederek görebilirsiniz. Eğer değilse yapı yaratır ve onu eklersiniz.

Her ne zaman bazı linkleri(Örneğin **PERSON**, **FATHER**, **MOTHER**, **FATHER_OF** ve **MOTHER_OF**) yaratmaya ihtiyacı olan satırı işlersiniz, ilk olarak linkin var olup olmadığını ve doğruluğunu kontrol edersiniz. Eğer doğru değilse hata alırsınız. Eğer doğruysa bir şey yapmayın. Eğer linkler yoksa aile ve çocuğun her ikisi için de onu yaratmalısınız.

Bu aşamayı yaptığınızda, red-black tree içinde tüm insanlarla sahip olacaksınız ve bütün insanlar aileleri ve çocukları için doğru linklere sahip olacaklar.

2. Dizi için grafik testi. Eğer ağaçta dizi varsa grafik karışır. Başka bir deyişle eğer bir kişi kendi ata ya da torunu olabilirse, burada bir problem vardır. Dizi için test CS302 de öğrenmiş olmanız gereken basit bir ilk derinlik aramadır. Test için eğer bir kişi onun kendi torunuysa takip eden sözde kod çalışacaktır.

/*kişi yapısında "visited" adında integer alan olduğunu varsayın, ve bu alan tüm insanlar için sıfırdan başlatılır. */

is_torun(Kişi *p)

{

```
    if (p->visited == 1) return 0;  /*Örneğin:bu kişiyi önce işleriz ve o
tamamlanır*/

    if (p->visited == 2) return 1;    /*Örneğin: dizi karıştırıldı.*/

    p->visited = 2;

    for all children of p do {
       if is_torun(cocuk) return 1;
    }
    p->visited = 1;
    return 0;
   }
```

Diziyi yazdırın. Eğer lab ın %20 nin sonunda vazgeçmek istiyorsanız, diğer insanların insan ağaçlarını yazdırın ve insan ağaçlarını dikkatle inceleyin. Aksi takdirde, bu genişlik öncelikli arama türüdür. Yeni bir kuyruk yaratmak istiyorsanız (toprint adı verilen bir Dllist olduğunda, öncelikle toprint içine tüm insanları koyabilirsiniz. Yada aile olamyan tüm insanları koyabilirsiniz.

Sonra bunun devamında bir döngü çalıştırın:

/* kişi yapısında "visited" adında integer alan olduğunu varsayın, ve bu alan tüm insanlar için sıfırdan başlatılır.*/

```
      while toprint is not empty
        take p off the head of toprint
        if p has not been printed, then
```

```
    if p doesn't have parents, or if p's parents have been printed
then
      print p
      for all of p's children, put the child at the end of doprint
    end if
  end if
end while
```

Ayrıca burada bu lab için makefile vardır. **http:// www.cs.utk.edu/~cs360/huang/lab1/**