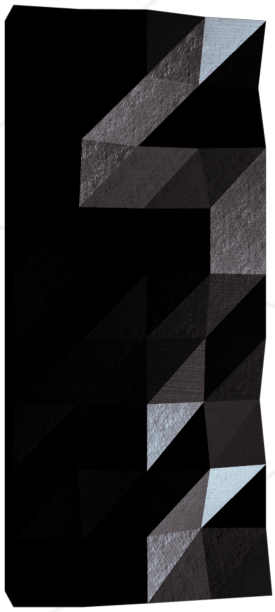

Minitalk

Özet:
Bu projenin amacı, UNIX sinyallerini kullanarak küçük bir veri değişim programını kodlamaktır

Versiyon: 2

İçindekiler

Bölüm I

Önsöz

(Z)-3-heksen-1-ol ve yaprak alkolü olarak da bilinen cis-3-Hexen-1-ol, taze kesilmiş yeşil çimen ve yaprakların yoğun çimenli yeşili kokusuna sahip renksiz yağlı bir sıvıdır.

Çoğu bitki tarafından küçük miktarlarda üretilir ve birçok yırtıcı böcek için çekici görevi görür. cis-3-Hexen-1-ol meyve ve sebze aromalarında ve parfümlerde kullanılan çok önemli bir aroma bileşiğidir.

Yıllık üretim yaklaşık 30 tondur.

Bölüm II

Genel Talimatlar

- Projeleriniz C programlama dilinde yazılmalıdır.
- Projeleriniz Norm'a uygun olarak yazılmalıdır. Bonus dosyalarınız/fonksiyonlarınız varsa, bunlar norm kontrolüne dahil edilir ve bu dosyalarda norm hatası varsa `0` alırsınız.
- Tanımlanmamış davranışlar dışında sizin fonksiyonlarınız beklenmedik bir şekilde sonlanmamalıdır (Segmentasyon hatası, bus hatası, double free hatası, vb.) . Eğer bunlar yaşanırsa s `0` alırsınız.
- Heap'de ayırmış olduğunuz hafıza adresleri gerekli olduğu durumlarda serbest bırakılmalıdır. Hiçbir istisna tolere edilmeyecektir.
- Eğer verilen görev `Makefile` dosyasının yüklenmesini istiyorsa, sizin kaynak dosyalarınızı `-Wall`, `-Wextra` , `-Werror`, flaglarini kullanarak derleyip çıktı dosyalarını üretecek olan Makefile dosyasını oluşturmanız gerekmektedir. Makefile dosyasını oluştururken cc kullanın ve Makefile dosyanız yeniden ilişkilendirme yapmamalıdır (relink).
- `Makefile` dosyanız en azından `$(NAME)`, `all`, `clean`, `fclean` ve `re` kurallarını içermelidir.
- Projenize bonusu dahil etmek için Makefile dosyanıza `bonus` kuralını dahil etmeniz gerekmektedir. Bonus kuralının dahil edilmesi bu projenin ana kısmında kullanılması yasak olan bazı header dosyaları, kütüphaneler ve fonksiyonların eklenmesini sağlayacaktır. Eğer projede farklı bir tanımlama yapılmamışsa, bonus projeleri `_bonus.{c/h}` dosyaları içerisinde olmalıdır. Ana proje ve bonus proje değerlendirmeleri ayrı ayrı gerçekleştirilmektedir.
- Eğer projeniz kendi yazmış olduğunuz `libft` kütüphanesini kullanmanıza izin veriyorsa, bu kütüphane ve ilişkili `Makefile` dosyasını proje dizinindeki `libft` klasörüne ilişkili `Makefile` dosyası ile kopyalamanız gerekmektedir. Projenizin `Makefile` dosyası öncelikle `libft` kütüphanesini kütüphanenin `Makefile` dosyasını kullanarak derlemeli ardından projeyi derlemelidir.
- Test programları sisteme yüklenmek zorunda değildir ve puanlandırılmayacaktır. Buna rağmen test programları yazmanızı şiddetle önermekteyiz. Test programları

sayesinde kendinizin ve arkadaşlarınız projelerinin çıktılarını kolaylıkla gözlemleyebilirsiniz. Bu test dosyalarından özellikle savunma sürecinde çok faydalanacaksınız. Savunma sürecinde kendi projeleriniz ve arkadaşlarınızın projeleri için test programlarını kullanmakta özgürsünüz.

- Çalışmalarınız atanmış olan git repolarına yüklemeniz gerekmektedir. Sadece git reposu içerisindeki çalışmalar notlandırılacaktır. Eğer Deepthought sizin çalışmanızı değerlendirmek için atanmışsa, bu değerlendirmeyi arkadaşlarınızın sizin projenizi değerlendirmesinden sonra gerçekleştirecektir. Eğer Deepthought değerlendirme sürecinde herhangi bir hata ile karşılaşılırsa değerlendirme durdurulacaktır.

Bölüm III

Proje talimatları

- Executable dosyalarınızı `client` ve `server`. olarak adlandırın .
- Hataları iyice ele almalısınız. Programınız hiçbir şekilde beklenmedik şekilde kapanmamalıdır (segmentation fault, bus error, double free vb.).
- Programınızda **memory leaks** olmamalıdır.
- **Program başına bir global değişkeniniz** olabilir (biri client , diğeri server için), ancak gerekli yerlerde kullanmanız gerekecektir.
- Zorunlu kısmı tamamlamak için aşağıdaki işlevleri kullanmanıza izin verilir:
  - `write`
  - `ft_printf` ve sizin kodladığınız herhangi bir eşdeğer kod
  - `signal`
  - `sigemptyset`
  - `sigaddset`
  - `sigaction`
  - `kill`
  - `getpid`
  - `malloc`
  - `free`
  - `pause`
  - `sleep`
  - `usleep`
  - `exit`

Bölüm IV

Zorunlu Bölüm

Client ve **server** şeklinde bir iletişim program oluşturmalısınız.

- İlk olarak server başlatılmalıdır. Başlatıldıktan sonra `PID`'sini. yazdırması gerekir.
- The client takes two parameters:
  - Server `PID`.
  - Gönderilecek string.
- Client, parametre olarak iletilen String'i server'a göndermelidir.
  String alındıktan sonra server'in onu yazdırması gerekir.
- Server'in string'i oldukça hızlı bir şekilde görüntülemesi gerekir. Çok uzun sürdüğünü düşünüyorsanız muhtemelen çok uzundur.

`100 karakter görüntülemek için 1 saniye çok fazla!`

- Server, yeniden başlatmaya gerek kalmadan arka arkaya birkaç client'dan dizeler alabilmelidir.
- Client ve server arasındaki iletişim **yalnızca** UNIX sinyalleri kullanılarak yapılmalıdır.
- Yalnızca bu iki sinyali kullanabilirsin: `SIGUSR1` ve `SIGUSR2`.

`Halihazırda bu türden bekleyen sinyalleriniz olduğunda, Linux sistemi sinyalleri kuyruğa KOYMAZ! Bonus zaman?`

Bölüm V

Bonus bölüm

Bonus liste:

- Server, client'a bir sinyal göndererek alınan her mesajı onaylar.
- Unicode karakter desteği!

Bonus kısım, yalnızca zorunlu kısım MÜKEMMEL ise değerlendirilecektir.Mükemmel, zorunlu kısmın entegre olarak yapıldığı ve arızasız çalıştığı anlamına gelir. TÜM zorunlu gereksinimleri geçmediyseniz, bonus bölümünüz hiç değerlendirilmeyecektir.

Bölüm VI

Teslim Etme ve Değerlendirilme

Her zamanki gibi Git repository'nize ödevinizi teslim edin. Savunma sırasında sadece deponuzdaki çalışmalar değerlendirilecektir. Doğru olduklarından emin olmak için dosyalarınızın adlarını iki kez kontrol etmekten çekinmeyin.