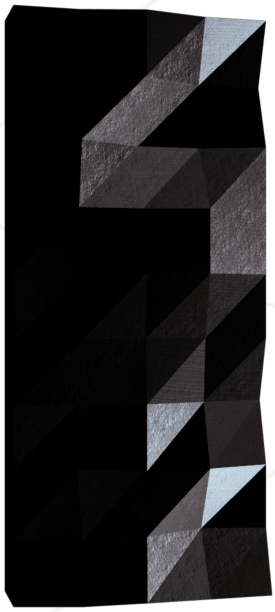

Get Next Line

Fd de bir satır okumak çok sıkıcı...

Özet: Bu projenin amacı sizin file descriptordan okunan bir satırı dönen fonksiyonu gerçeklemenizi sağlamaktır.

Versiyon: 10

İçindekiler

Bölüm I

Amaçlar

Bu proje sadece çok kullanışlı bir fonksiyonun koleksiyonunuza eklemenize yardımcı olmayacak aynı zamanda size `C` programlamada çok dikkat çekici ve yeni bir konsept olan "static variables" i öğrenme fırsatı verecektir.

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

Zorunlu kısım - Get_next_line

Fonksiyon adı	`get_next_line`
Prototip	`char *get_next_line(int fd);`
Teslim edilecek dosyalar	`get_next_line.c, get_next_line_utils.c, get_next_line.h`
Parametreler	`Okuma işleminin yapılacağı file descriptor`
Return değeri	`Read line: correct behavior` `NULL: okunacak bir şey yoksa veya bir hata ile karşılaşılmışsa`
Harici fonksiyonlar	`read, malloc, free`
Açıklama	`file descriptordan okunan bir satırı dönen bir fonksiyon yazın`

- Bir döngü içerisinde sizin yazmış olduğunuz `get_next_line` fonksiyonunun çağırılması ile döngü sonuna kadar file descriptordan her döngüde bir satır okumanız gerekmektedir.
- Fonksiyonunuz en son okunmuş satırı dönmelidir. Eğer okunacak bir şey kalmamışsa veya bir hata ile karşılaşılmışsa NULL değerini dönmelidir.
- Fonksiyonunuzun dosyadan veya standart inputtun okuduğu durumlarda uygun davrandığından emin olun.
- Bu projede `libft` kullanmanız yasaktır. Sizin çalışmanıza `get_next_line` çalışmanızdaki fonksiyonları içeren `get_next_line_utils.c` dosyasını eklemeniz gerekmektedir.
- Programınız sizin `get_next_line` projenizde okuma çağrılarında buffer size olarak kullanılacak olan `-D BUFFER_SIZE=xx` flagi ile compile edilmelidir. Bu değer sizi değerlendirenler tarafından ve moulinette tarafından modifiye edilecektir.
- Program şu şekilde compile edilecektir:
`cc -Wall -Wextra -Werror -D BUFFER_SIZE=42 <files>.c`.
- Sizin `read` metodunuz bir dosyadan veya stdinden okuma yapmak için compilation

sırasında tanımlanmış olan BUFFER_SIZE değişkenini mutlaka kullanmalıdır. Bu değer değerlendirme sürecinde test amaçları için değiştirilecektir.

- `get_next_line.h` header dosyasında en azından fonksiyonunu prototipi olmalıdır.

Sizin fonksiyonunuz BUFFER_SIZE değeri 9999, 1 veya 10000000 iken de hala çalışıyor mu? Nedenini biliyor musunuz?

Her get_next_line fonksiyonu çağırıldığında olabildiğince az okuma yapmanız gerekmektedir. Eğer yeni satır karakteri ile karşılaşırsanız, şimdi bulunduğunuz satırı dönmeniz gerekmektedir. Bütün dosyayı bir anda okuyup satır satır process etme işini yapmayın.

Projeleriniz test etmeden teslim etmeyiniz. Çalıştırabilecek çok fazla test vardır, base dosyalarınızı kontrol edin. Dosyadan, redirection ve standart inputtan okumayı deneyin. Standart outputtan newline karakteri gönderdiğinizde programınız nasıl davranıyor? Ve CTRL-D?

- `lseek` kullanmanıza izin verilmemektedir. Dosya okuma sadece bir kere yapılmalıdır.
- Eğer ilk fd birinci dosya bitmeden diğer dosyayı okumaya geçerse `get_next_line` fonksiyonunun tanımsız bir davranış sergilediğini düşünürüz.
- Son olarak binary dosyasından okuma yaparken `get_next_line` fonksiyonun tanımlanmamış davranış sergilediğini düşünürüz. Ancak eğer eğer isterseniz bu davranışı tutarlı hale getirebilirsiniz.
- Global değişken kullanmak yasaktır.
- Önemli: Döndürülen satırlar `'\n'` karakterini içermeledir. Eğer dosya sonuna geldiyseniz buna ihtiyaç yoktur.

Static variable'ın ne olduğunun detayına bakmak isterseniz aşağıdaki linki inceleyebilirsiniz.
https://en.wikipedia.org/wiki/Static_variable

Bölüm IV

Bonus Kısım

`get_next_line` projesi çok açık ve basittir bundan dolayı bonus için yapacak çok az şey kalmıştır ama ben sizin bir sürü yaratıcı fikriniz olduğuna inanıyorum. Eğer zorunlu kısmı tamamladıysanız burayı da yapmayı deneyebilirsiniz. If zorunlu kısımda hata varsa bonus kısım değerlendirmeye alınmayacaktır.

Bu bölüm için `_bonus.[c\h]` ile sonlanan üç dosyayı teslim edin

- Başarılı olabilmek için `get_next_line` fonksiyonunu sadece bir tane static variable kullanarak gerçekleyin.
- `get_next_line` fonksiyonunda birden fazla fd'yi yönetebilmelisiniz. Örneğin eğer 3,4,5 file descriptorları okuma için uygunsa, ardından `get_next_line` fonksiyonunu bir kez 3'te bir kez 4'te bir kez daha 3'te ardından bir kez daha 5'te çağırabilirsiniz. Bunu yaparken file descriptorlardaki reading threadi kaybetmemeniz gerekmektedir.