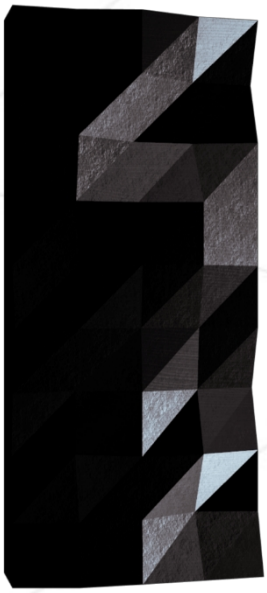

미니토크

요약:

이 프로젝트의 목적은 UNIX 신호를 사용하여 소규모 데이터 교환 프로그램을 코딩하는 것입니다.

버전: 3

콘텐츠

1장 머리말

(Z)-3-헥센-1-올 및 잎 알코올로도 알려진 시스-3-헥센-1-올은 갓 자른 푸른 풀과 잎에서 나는 강렬한 풀잎 냄새가 나는 무색의 유성 액체입니다.

대부분의 식물에서 소량 생산되며 많은 포식성 곤충의 유인제로 작용합니다. cis-3-헥센-1-올은 과일 및 채소 향료와 향수에 사용되는 매우 중요한 향기 화합물입니다.

연간 생산량은 약 30톤입니다.

제2장

일반적인 지침

- 프로젝트는 C로 작성해야 합니다.
- 프로젝트는 규범에 따라 작성되어야 합니다. 보너스 파일/함수가 있는 경우 규범 검사에 포함되며 내부에 규범 오류가 있는 경우 0을 받게 됩니다.
- 정의되지 않은 동작을 제외하고는 함수가 예기치 않게 종료(세그먼트 오류, 버스 오류, 더블 프리 등)되어서는 안 됩니다. 이런 일이 발생하면 프로젝트가 작동하지 않는 것으로 간주되어 평가 중에 0점을 받게 됩니다.
- 할당된 모든 힙 메모리 공간은 필요할 때 적절히 해제되어야 합니다. 어떠한 누수도 용납되지 않습니다.
- 주제에서 요구하는 경우 소스 파일을 필요한 출력으로 컴파일하는 메이크파일을 -Wall, -Wextra 및 -Werror 플래그와 함께 제출해야 하며, cc를 사용하고, 메이크파일은 리링크되지 않아야 합니다.
- 메이크파일에는 최소한 $(NAME), all, clean, fclean 및 re.
- 프로젝트에 보너스를 추가하려면 프로젝트의 주요 부분에 금지된 다양한 헤더, 라이브러리 또는 함수를 모두 추가하는 규칙 보너스를 메이크파일에 포함해야 합니다. 보너스는 주제에서 다른 것을 지정하지 않은 경우 다른 파일 _bonus.{c/h}에 있어야 합니다. 필수 부분과 보너스 부분 평가는 별도로 진행됩니다.
- 프로젝트에서 라이브러리 사용을 허용하는 경우 해당 소스와 관련 메이크파일을 관련 메이크파일이 있는 라이브러리 폴더에 복사해야 합니다. 프로젝트의 메이크파일을 사용하여 라이브러리를 컴파일한 다음 프로젝트를 컴파일해야 합니다.

- 이 **작업은 제출할 필요가 없고 채점되지** 않더라도 프로젝트에 대한 테스트 프로그램을 만드는 것이 좋습니다. 이렇게 하면 자신의 작업과 동료의 작업을 쉽게 테스트할 수 있습니다. 이러한 테스트는 방어하는 동안 특히 유용합니다. 실제로 방어하는 동안에는 자신의 테스트 및/또는 평가하는 동료의 테스트를 자유롭게 사용할 수 있습니다.

- 할당된 git 저장소에 작업을 제출합니다. git 저장소에 있는 작업만 채점됩니다. 작업을 채점하도록 Deepthought가 지정되면 다음과 같이 채점이 완료됩니다.

동료 평가가 끝난 후 딥씽킹이 채점하는 동안 작업의 어느 부분에서든 오류가 발생하면 평가가 중지됩니다.

제3장

프로젝트 지침

- 실행 파일의 이름을 클라이언트 및 서버로 지정합니다.
- 소스 파일을 컴파일할 메이크파일을 제출해야 합니다. 다시 링크해서는 안 됩니다.
- 리브프트를 확실히 사용할 수 있습니다.
- 오류를 철저하게 처리해야 합니다. 어떤 경우에도 프로그램이 예기치 않게 종료되어서는 안 됩니다(세분화 오류, 버스 오류, 더블 프리 등).
- 프로그램에 **메모리 누수가** 없어야 합니다.
- **프로그램당 하나의 전역 변수를** 가질 수 있지만(클라이언트용과 서버용 각각 하나씩), 사용을 정당화해야 합니다.
- 필수 부분을 완료하기 위해 다음 기능을 사용할 수 있습니다:
    - 쓰기
    - ft_printf 및 이에 상응하는 모든 코딩
    - 신호
    - sigemptyset
    - 시그애드셋
    - 시그액션
    - kill
    - getpid
    - malloc

◦ 무료

◦ 일시 중지

◦ 수면

◦ 수면

◦ exit

4장 필수 부분

클라이언트와 **서버의** 형태로 통신 프로그램을 만들어야 합니다.

- 서버를 먼저 시작해야 합니다. 서버가 시작되면 PID를 인쇄해야 합니다.
- 클라이언트는 두 개의 매개 변수를 사용합니다:
  - 서버 PID입니다.
  - 전송할 문자열입니다.
- 클라이언트는 매개변수로 전달된 문자열을 서버로 전송해야 합니다. 문자열이 수신되면 서버는 이를 인쇄해야 합니다.
- 서버는 문자열을 매우 빠르게 표시해야 합니다. 빠르다는 것은 너무 오래 걸린다고 생각되면 너무 길다는 뜻입니다.

100자를 표시하는 데 1초는 **너무 길어요**!

- 서버는 다시 시작할 필요 없이 여러 클라이언트로부터 연속으로 문자열을 수신할 수 있어야 합니다.
- 클라이언트와 서버 간의 통신은 UNIX 신호만을 사용하여 수행해야 합니다.
- 이 두 신호만 사용할 수 있습니다: SIGUSR1과 SIGUSR2.

Linux 시스템은 이미 이러한 유형의 대기 중인 신호가 있는 경우 신호를 대기열에 넣지 않습니다! 보너스 시간?

5장 보너스 부분

보너스 목록:

- 서버는 클라이언트에 신호를 다시 전송하여 수신된 모든 메시지를 확인합니다.
- 유니코드 문자 지원!

보너스 부분은 필수 부분이 완벽한 경우에만 평가됩니다. 완벽하다는 것은 필수 부분을 완벽하게 완료하고 오작동 없이 작동한다는 의미입니다. 모든 필수 요건을 통과하지 못한 경우 보너스 부분은 전혀 평가되지 않습니다.

제6장

제출 및 동료 평가

평소처럼 Git 저장소에 과제를 제출하세요. 방어 중에는 저장소 내의 작업만 평가됩니다. 파일 이름이 올바른지 다시 한 번 확인하여 주저하지 마세요.

file.bfe:VAAe8ElCrUAbXivz0ueiIpv/u/ia9PL50+HI+8/bgPKLESHlp
tPLpu0PW9zWV/LwDVaOqCRCGu6Gopk1X0i6Kn7t