김현수 (hscornelia) 수정

팔로우 중인 프로필 전체 프로필 보기

돌소리
알립니다
구인구직란
18학번 모임
버그 제보 및 기능 제안
김현수
족보

개인정보 처리방침

2020-2 System Programming 시스템 프로그래밍 에거 교수님 중간고사 필기 / 실기 족보 일 년 전

이인용

2020 2학기 시스템 프로그래밍 (에거 교수님) 중간고사 필기

1. 16개 객관식 (각 0.5점) 여러 개 있을 수 있으니 여러 개면 여러 개 고르세요

보기는 안 적었습니다.

1. 시스템 콜은 어디로 리턴하는가 i*curr?* *i*next? abort? ??
2. 페이지 아웃된 페이지 접근해서 폴트가 났다 어디로 리턴하는가
3. malloc() 호출할 때 프로세서 모드는?
4. 타이머 인터럽트는 싱크? 어싱크?
5. /o은 싱크? (뭐라고 적은 거지)
6. system call argument 패스 방식 (스택? 레지스터? 중에 뭐냐)
7. 프로세스 끝난 거 뭐라고 부름 (언데드? 좀비? 중에 뭐냐)
8. 6개의 함수 중에서 콜 성공했을 때 리턴하지 않는 함수들은? (fork(), execv(), exit(), open(), write() 등등)
9. 6개의 함수 중(아까랑 같음)에서 성공했을 때 2번 리턴하는 함수는?
10. 리눅스 x86_64에서 시스템 콜하는 인스트럭션은?
11. 익명 파이프로 소통하지 못 하는 관계 1) 부모-자식 2) 형제 자매 3) 조부모-자식 4) 관계없는
12. the process receive signals ... 1) ? 2) ? 3) 최대한 빨리 4) 다음 인터럽트 때 ㅈㅅ
13. 같은 타입의 시그널이 여러 번 왔다. 이건 큐에 쌓이는가? 1) yes 2) no
14. 버퍼 I/O인 것은? 1) buffered I/O 2) standard I/O 3) unix I/O 4) 까먹음

15. UNIX 누가 만들었나
16. 0.5점 받고 싶음? 네!

2. file descriptor와 fork, dup 등

2개의 코드가 있는데
C1은 `fd[0] = open(~); fd[1] = open(~);` 했고 (같은 파일)
C2는 `fd[0] = open(~); fd[1] = dup(fd[0]);` 했음

```
master = getpid();
// C1 C2 차이 코드
for (int i = 0; i < 2; i++) {
  pid = fork();
    read(fd[pid ? 0 : 1], &c, 1);
}
if (master == getpid()) {
  read(fd[0], &c, 1);
    write(STDOUT_FILENO, &c, 1);
}
```

파일 내용이 `0123456789`일 때 C1과 C2의 출력과 왜 그렇게 나오는지 설명하세요.

3. 리눅스 커널이 VM을 통해 fork()를 최적화한 방법

리눅스 커널이 VM의 features를 이용하여 fork()를 효율적으로 구현한다. 이에 대해 상세하게 적으세요.

4. fork() 장난질

대략 아래와 같은 코드가 주어짐. (대충 적음)

```
pid = getpid();
int i;
for (i =0; i < 4; i++) {
  printf("%c%d\n", pid > 0 ? 'p' : 'c', i);
  pid = fork();
  if (i % 2 == 0) {
    if (pid == 0) break;
  }
  else {
    if (pid > 0) break;
  }
}
if (pid > 0) waitpid(pid, NULL, 0);
printf("b%d\n", i);
```

A) 생성되는 프로세스를 그리고 걔네가 뭘 프린트하는지 쓰세요.
B) 프린트 순서가 어떻게 되는지, 가능한 경우의 개수는?

ㅁㄴㅇㄹ

5. Implicit free list

헤더와 푸터가 있는 블록? 시스템을 가진다고 할 때 boundary tag가 다음과 같다.

```
[      size      | a b c ]
```

a 비트는 항상 0,
b 비트는 가까운 블록이 allocated 면 1 아니면 0,
c 비트는 이 블록이 allocated 면 1 아니면 0을 의미한다.
여기서 가깝다는 건 next higher address를 의미

메모리 구조가 아래와 갖고 가운데 친구를 free 했을 때 메모리가 어떻게 되는지 적으시오. 같으면 same이라고 쓰고 안 쓴 건 안 푼 거임

높은 주소
```
[ 0x00000012 ]
[ 0x20121211 ]
[ 0x03010203 ]
[ 0x00000012 ]
[ 0x00000011 ]
[ 0x00000011 ] <- 여기 free함
[ 0x00000011 ]
[ 0x00000011 ]
[ 0x00000013 ]
[ 0x19980522 ]
[ 0x19850916 ]
[ 0x00000013 ]
```
낮은 주소

6. explicit free list

이건 못 베껴옴 ㅋㅋ

address ordering 인 explicit free list 가 있고 1비트가 allocate 4비트부터 사이즈인 헤더 푸터 가진 구조.
[프리 | 알록 | 프리 | **알록** | 알록 | 프리 | 알록] 그림이 있는데 (시궈스는 틀렸을 수 있음)
볼드 처리된 부분을 프리 했을 때 메모리 내용들이 어떻게 변하는지 적으세요.

실기

폴더 여러 개 입력 받고 그 중 duplicated된 파일들이 얼마나 있는지 통계를 보여주는 프로그램 만들기

- 단계 별로 진행되며 뒤 단계는 앞 단계를 완성해야 이어서 구현할 수 있다
- make init 을 하면 이전 단계 (A의 경우 initial 소스)가 복사된다. 여러 번 하면 큰일 남(?)
- 각 단계 별로 힌트들이 있다. 힌트 구성은 대충 1) 설명을 볼 건지 2) 수도 코드를 볼 건지 3) 필요한 시스템 함수들을 볼 건지 4) 정답을 볼 건지 같은 방식이고 받으면 점수가 깍인다. 예를 들어 C는 40점 만점인데 2번과 3번을 선택하면 -20점, 정답을 보면 -35점이다. 아무튼 D를 풀기 위해서는 C를 받고 넘어가는 편이 좋을 듯
- 적당한 테스트 세트를 줬고 `make test`로 짧은 것 실행 가능. doxygen 으로 된 문서를 찬찬히 읽어봐야 함 (스펙 다 적혀있음) 링크 파일은 제외할 건지 등등
- 해시 리스트를 관리하기 위한 자료 구조와 관련 메소드는 주어짐. 이것도 doxygen 문서를 보면 사용할 수 있음

A 10점

메인 함수 짜기. argv[] 를 순회하면서 B의 `process_dir()` 를 호출하고 D의 `statistics()` 호출하기

B 40점

dir path를 받고 그 안의 파일들을 순회하면서 처리하기
이 때 C의 md5sum() 을 호출하여 각 파일 해시 리스트에 추가하기

C 40점

md5sum() 함수 구현하기. md5sum 작업은 리눅스에 있는 /usr/bin/md5sum 을 사용하세요.

D 30점

순회하면서 얻은 list 를 포맷에 맞게 통계로 보여주기

실기 정답(?)

▶ 각 단계 정답

첨언

쉬웠다! 잘 봤다는 건 아님 ㅋㅋ
필기 5시 5분 시작 6시 20분 끝
실기 7시 시작 9시 종료.. 였으나 준비 미흡으로 학생들에게 개인 비밀번호를 주는 데 시간을 잡아먹어 7시 40분경 시작해서 9시 30분 종료로 변경, 그리고 30분 연장을 두 번 해서(학생들이 원해서) 10시 30분에 끝났다.

추천(0)

▶ 시스템프로그래밍 × 태그추가

ㄴ 확인