1. Exercise 9.15 on page 915 with the following

Malloc(5) 16 0x11

Malloc(8) 16 0x11

Malloc(23) 32 0x21

Malloc(35) 40 0x29

1. Exercise 9.19 on page 916

1.

정답 : (a) In a buddy system, up to 50% of the space can be wasted due to internal fragmentation

(b) first-fit은 대체적으로 best-fit보다 빠름

(c):constant time이다. 메모리 주소의 정렬 방법에 상관없이

(d)외부단편화와도 관련이 있다.

2.

정답 : (d) using the first-fit algorithm on a free list that is ordered according to decreasing block sizes results in low performance forallocations, but avoid external fragmentation

(a):할당시에 빠른 성능을 내지만 외부단편화 발생

(b):메모리 주소가 증가하는 순으로 정렬될 필요 없음

(c):가장 적절한 free-block을 선택한다.

3.

(b) ma가&sweep garabage collectors are called conservative if they treat everything that looks like a pointer as a pointer

1. Explain the role of the following functions
2. Getaddressinfo : 호스트 이름, 호스트 주소, 서비스 이름을 socket 주소의 구조체로 convert한다. Host와 service가 주어지면, 소켓 주소 구조체의 연결리스트를 반환한다.

b. socket : 프로세스에서 network i/o를 하기위한 통로를 open하는 것이다. Ip와 port를 지정받는다.

c. bind : 서버 프로그램이 well-known port를 이용할 수 있도록 연결하는 함수이다. Ip:e-port를 ip:well-known port로 변경한다..

d. connect : 클라이언트가 connect함수를 호출해 서버와 연결할 수 있도록 요청하는 함수이다.

e. listen : active한 socket을 외부로부터 요청을 받아수행할 수 있도록 passive한 상태로 만드는 함수이다.

f. accept : accpet함수는 client의 연결 요청을 listen fd에서 기다리고, 요청이 도달할 경우 connection fd를 만들어 return하는 함수이다.

1. Enumerate the difference between process and thread concisely

Process : data registers, condition codes, stack pointer, program counter등의 register에서 관리 하는 부분과 virtual memory, descriptor table를 갖는 kernel context와 code ,data, and stack 등의 program visible state가 모두 존재한다.

Thread: process와 마찬가지로 register에서 관리하는 부분들과 stack이 존재하지만 program visible state와 kernel context에 해당하는 영역이 존재하지 않는다.

1. Exercise 12.6 on page 1031

|  |  |  |  |
| --- | --- | --- | --- |
|  | Main thread | Peer thread 0 | Peer thread 1 |
| Ptr | y | Y | Y |
| cnt | N | Y | y |
| i.m | Y | N | N |
| Msgs.m | Y | Y | y |
| Myid.p0 | N | Y | N |
| Myid.p1 | N | N | Y |

1. Exercise 12.17 on page 1067

a.main thread가 peer thread가 message를 출력하기 전 exit를 call하기 때문이다. Exit는 모든 thread를 terminate시키기에 오류가 발생한다.

b. exit 함수를 pthread\_exit나 pthread\_join으로 바꾼다. Pthread join은 peer thread의 실행이 끝날 때까지 기다려준다.

혹은 return을 쓰면 current thread만 종료되니 return도 사용가능하다.

1. Exercise 12.18 on page 1068

a. unsafe

b. safe

c. unsafe