1. 커널이란?

-컴퓨터 운영체계의 가장 중요한 핵심으로 운영체계의 다른 모든 부분에 여러 가지 기본적인 서비스를 제공한다.

1. 시분할 시스템이란?

-운영체제에서 CPU 스케줄링과 다중 프로그래밍을 이용해서 각 사용자들에게 컴퓨터 자원을 시간적으로 분할하여 사용할 수 있게 해준다.

1. 캐시란?

-속도가 느린 기억장치와 속도가 빠른 기억장치 중 최근에 사용한 일부를 넣어두거나 가까운 데이터를 넣어놔서 속도 차이를 보완하기 위한 장치이다.

1. 저장장치 계층 구조

레지스터 – 캐시 – 메모리 – 하드 디스크

1. Write-back Cache와 Write-Through Cache의 차이점.

<http://melonicedlatte.com/computerarchitecture/2019/02/12/203749.html>

Write-Through Cache : 데이터가 캐시 됨과 동시에 주기억장치 또는 디스크에 기입된다.

장점 : 캐시와 메모리에 업데이트를 같이 하여, 데이터가 안정적

Write-Back Cache : 캐시 내에 일시적으로 저장된 후에 블록 단위에 캐시로부터 해제되는 때에 주기억장치 디스크에 저장된다.

장점 : Through방식에 비해 속도가 빠름

1. 프로세스란?

하드에 있는 프로그램이 메모리에 올라와서 실행상태인 프로그램을 얘기함

1. Thread란?

프로세스를 여러가지 작업(Task)로 나눈 것을 스레드라고 함.

1. 프로세스의 상태 4가지

준비 , 실행, 대기 , 종료

1. Context switching

<https://nesoy.github.io/articles/2018-11/Context-Switching>

-현재 진행하고 있는 Task(Process,Thread)의 상태를 저장하고 다음 진행할 Task의 상태 값을 읽어 적용하는 과정을 말함.

1. 가비지 콜렉터란?

<https://12bme.tistory.com/57>

메모리 누수 문제를 해결하기위해 더 이상 참조하지 않는 쓸모 없는 객체,데이터를 자동으로 제거해주는 기능이다.

1. Process와 Thread의 차이점

Thread는 프로세스와 달리 Stack을 제외한 영역을 공유하므로 Context Switching이 빠르다.

1. 멀티 프로세스와 멀티 스레드의 차이

위와 동일

1. Synchronous interrupt와 asynchronous interrupt 차이

<https://jaejin0me.github.io/post/posts126/>

Synchronous interrupt : CPU의 제어 유닛이 명령어를 실행하는 과정에서 발생하는데, 제어 유닛은 명령어 실행을 마친 후에만 이를 발생시키기 때문에 동기적이라고 한다.

asynchronous interrupt : 다른 하드웨어 장치가 CPU 신호와 관계없이 아무 때나 발생한다.

1. 선점형 스케줄링과 비선점형 스케줄링

-선점형은 다른 프로세스에게 CPU점유권을 뺏어오는 것이 가능

-비선점형은 뺏어오는 것이 불가능하다.

1. CPU 스케줄링이란?

여러 프로세스가 CPU 점유권을 순차적으로 점유해 CPU사용량을 최대한 늘리기 위함

1. CPU를 효율적으로 향상시켰나 판단하는 기준 4가지
2. FCFS
3. SJF
4. Round Robin
5. HRRN
6. Aging 기법
7. SRTF
8. Critical section

-임계구역 : 동기화 방법중 하나로 공유 자원의 독점을 보장해주는 역할을 수행한다.

예를 들어 코드상에 공유자원을 변경하는 코드가 적혀있는 부분이다.

1. Critical section 문제 해결 3가지

<https://swalloow.tistory.com/70>

1. Semaphore, Mutex
2. Monitor, 인터페이스만 제공하는 것
3. Deadlock

<https://jwprogramming.tistory.com/12>

프로세스가 자원을 얻지 못해 다음 처리를 하지 못하는 상태로 “교착 상태”라고도 하며 시스템적으로 한정된 자원을 여러 곳에서 사용하려고 할 때 발생합니다.

1. Deadlock필요조건 4가지

상호배제 : 자원은 한 번에 한 프로세스만 사용가능하다.

점유 대기 : 최소한 하나의 자원을 점유하고 있으면서 다른 프로세스에 할당되어 사용하고 있는 자원을 추가로 점유하기 위해 대기하는 프로세스가 있어야한다.

비선점 : 다른 프로세스에 할당된 자원은 사용이 끝날 때까지 빼앗을 수 없어야한다.

순환 대기 : 프로세스의 집합에서 순환적(원형)으로 각 필요한 자원을 다른 프로세스가 점유하고 있다.

1. Deadlock해결방법 3가지
2. Memory Overlay
3. 메모리 배치 정책

<https://m.blog.naver.com/PostView.nhn?blogId=sungeuns&logNo=50098110493&proxyReferer=https%3A%2F%2Fwww.google.com%2F>

고정분할 : 메모리를 고정된 크기의 파티션으로 분할

가변분할 : 각 프로세스에게 프로세스가 요구하는 정확한 메모리 크기를 할당

1. 가상 메모리 시스템이란?
2. 절대주소와 논리주소의 차이
3. 페이지 교체란?
4. FIFO
5. LRU
6. LFU
7. NUR
8. 집약성이란? 시간집약성과 지역집약성
9. 캐시 알고리즘 – 집약성을 이용
10. Threshing – 하드입출력이 잦아져 정지된것과 같음
11. Disk Scheduling – SSTF
12. SCAN
13. C-SCAN
14. DLL이란?
15. DFS와 BFS의 차이
16. 스택이 사용되는 경우

-실행 취소, 역순 문자열, 후위표기법, 인터럽트

1. 큐가 사용되는 경우

-CPU 스케줄링

1. New 와 malloc의 차이

New는 키워드 malloc은 함수여서 malloc의 경우 헤더파일 추가와 void반환이여서 자료형과 자료형의 크기를 명시적으로 반드시 지정해줘야한다. New는 생성자를 자동으로 호출하는 반면 malloc은 불가능하다.

1. Heap 영역

프로그래머가 동적 할당/해제 하는 영역

1. Hash Table이란?

<https://velog.io/@cyranocoding/Hash-Hashing-Hash-Table%ED%95%B4%EC%8B%9C-%ED%95%B4%EC%8B%B1-%ED%95%B4%EC%8B%9C%ED%85%8C%EC%9D%B4%EB%B8%94-%EC%9E%90%EB%A3%8C%EA%B5%AC%EC%A1%B0%EC%9D%98-%EC%9D%B4%ED%95%B4-6ijyonph6o>

-Key와 Value로 이루어진 자료구조로 Hash함수를 사용하여 키를 해시 테이블 주소로 변환한다.

1. Hash 함수란?
2. Hash충돌시 해결 기법 2가지 (Chaining기법, Open Addressing 기법) 설명

Open Addressing : 비어있는 메모리 공간을 찾아가는 것으로

선형 탐색(+1)이나 제곱탐색, 이중 해시(해시를 한번 더 적용)등으로 저장한다.

Chaining : 해시테이블의 데이터의 뒤에 연결한다.

1. Smart Pointer란?

C++의 메모리 누수를 방지하기 위한 매커니즘으로 스마트포인터로 선언한 객체의 해제를 자동으로 해준다.

1. Unique\_ptr, Shared\_ptr, Weak\_ptr에 대해 설명하시오

Unique\_ptr : 하나의 스마트 포인터만이 특정 객체를 소유할 수 있도록 객체에 소유권 개념을 도입

Shared\_ptr : 하나의 특정 객체를 참조하는 스마트 포인터가 총 몇 개인지를 참조하는 포인터, 참조 횟수가 0이 되면 delete 키워드를 사용하여 메모리를 자동으로 해제

Weak\_ptr : Shared\_ptr 인스턴스가 소유하는 객체에 대한 접근을 제공하지만, 소유자의 수에 포함되지 않는 포인터입니다. 서로가 상대방을 가리키는 S\_P를 가지고 있다면, 참조 횟수는 절대 0이 되지 않으므로 메모리는 해제되지 않는 데 이런 것을 제거하기 위해서 사용됩니다.

1. 캡슐화란?

외부에서 조작을 대비해서 외부에서 특정 속성이나 메서드를 사용자가 사용할 수 없도록 숨겨놓은것입니다.

1. 오버로딩과 오버라이딩의 차이점
2. Static이 저장되는 영역은?
3. Template란?

-여러 자료형을 받아오거나, 다양한 자료형으로 코드를 작성할 때 여러 번 쓸 필요없도록 하기위해 하나의 키워드로 여러 자료형을 받아올 수 있습니다.

1. Virtual 키워드란 무엇인가?

-파생클래스에서 함수를 재정의 할 것으로 기대되는 부분에 작성하는 것으로 virtual키워드를 가진 클래스는 가상 테이블을 가리키는 가상 포인터를 가지게 됩니다.

1. Vtable, vptr에 대해서 설명하시오
2. 가상 소멸자란?
3. 순수 가상함수란?
4. 추상클래스란?
5. 인터페이스와 추상클래스의 차이점을 말하시오.
6. AVL 트리란?
7. 레드-블랙 트리의 특징
8. BST란?
9. Inorder, Preorder, Postorder 설명하시오
10. DDL, DML, DCL에 대해 설명하시오
11. MMU란?
12. 외부단편화와 내부단편화
13. OSI 7계층에 대해 설명하시오.
14. TCP/UDP 차이점
15. 3 way-handshake란?
16. Client/server와 p2p의 차이점
17. 랜더링 파이프라인
18. 짐벌락이란?
19. 쿼터니언이란?
20. 메모리 풀이란?

-new와 delete가 반복적으로 일어나면 프로그램 속도에 영향, 파편화 현상을 주기 때문에 미리 특정공간만큼을 확보해둔 후 미리 확보된 공간을 제공, 회수함으로써 프로그램 속도를 높이는 방법.

1. 업캐스팅과 다운캐스팅에 대해 설명하시오.
2. Dynamic\_cast란?

-runtime 시에 작동한다 즉 virtual function을 가지고 있는 클래스에 사용한다.

1. Vector와 list의 차이
2. reserve()와 resize()의 차이
3. 객체지향의 4대 특징이란?
4. 구조체와 클래스의 차이점을 설명하시오.

클래스는 private, protected가 가능하고 생성자를 호출한다.

1. 적과의 앞뒤 판별을 하는 방법
2. 적이 왼쪽에 있는지 오른쪽에 있는지 판별하는 방법.
3. 컴파일 과정을 쓰시오.
4. 전처리기가 헤더 파일 삽입과 매크로 치환
5. 컴파일러가 코드를 기계어로 만들어 오브젝트 파일을 만듭니다.
6. 링커가 라이브러리들을 가져오고 실행파일을 만들어줍니다.
7. CISC와 RISC의 차이점
8. 조명의 종류 글로벌 일루미네이션
9. 고정기능 파이프라인과 프로그래머블 파이프라인의 차이점
10. Stack을 Thread에서 독립적으로 할당하는 이유는?
11. Map과 hash-map의 차이점
12. Thread 동기화 방법에 대해 쓰시오.
13. Mutex와 Semaphore의 차이점
14. Swapping이란?
15. C++ erase와 remove의 차이
16. PBR이란?
17. 아스키 코드와 유니코드 차이점
18. []와 at의 차이
19. 다익스트라 알고리즘이란?
20. 싱글스레드와 멀티스레드의 차이점

멀티 스레드는 응답성이 좋고 자원공유를 효율적으로 할 수 있고 작업이 분리되어서 코드가 간결해진다.

1. 메모리 단편화 해결방법 5가지(압축,통합,페이징,세그멘테이션,메모리풀)
2. Culling 기법이란?
3. 광원의 종류
4. 텍스처 매핑이란?
5. UV 좌표란?
6. 쉐이더란?
7. Draw Call에 대해 설명하시오
8. TCP/IP
9. Prim 알고리즘과 Kruskal 알고리즘의 차이
10. Blocking communication과 non-blocking communication 차이
11. 부동소수점 연산
12. 두 개의 동전을 던지는 실험에서 앞면이 나오는 횟수를 X라 할 때 X의 확률 분포 P(X=x)를 구하여라.
13. 함수형 프로그래밍이란?
14. NP Problem
15. Topological sort와 minimum, spanning tree

네트워크

1. 트랜잭션
2. Rollback
3. commit

언리얼 문제

1. RPC란?
2. replicated란?
3. 언리얼의 Garbage collection에 대해 설명하시오
4. 리플렉션 이란?
5. 블루프린트란 무엇인가요?
6. UFUNCTION, UPROPERTY, UCLASS 사용 이유

Class A { A(){ cout << “1”; } virtual ~A(){ cout << “2”; } };

Class B { B(){ cout << “3”; } virtual ~B(){ cout << “4”; } };

B b;

위 구문의 예상되는 출력값을 서술하세요.

1. 디퍼드와 포워드
2. 반사벡터 공식