**<4과목: 프로그래밍 언어 활용>**

C 언어에서 문자열 처리 함수의 서식과 그 기능의 연결

strlen(s): s의 길이를 구한다

strcpy(s1, s2): s2를 s1으로 복사한다

strrev(s): s를 거꾸로 변환한다

틀린 것: strcmp(s1, s2): s1과 s2를 연결한다. 기능: 두 문자열을 비교한다

&&, || , !

and or 연산자, c언어는 0이 아닌 모든 값은 다 True, 1

IP 프로토콜에서 사용하는 필드와 해당 필드에 대한 설명

Header Length는 IP 프로토콜의 헤더 길이를 32비트 워드 단위로 표시

Time To Live는 송신 호스트가 패킷을 전송하기 전 네트워크에서 생존할 수 있는 시간을 지정한 것

Version Number는 IP 프로토콜의 버전번호를 나타낸다

틀린 것: Packet Length는 IP 헤더를 제외한 패킷 전체의 길이를 나타내며 최대크기는 2^32 -1 비트이다 (헤더 포함, 16비트, 최대값 2^16 - 1)

RIP 라우팅 프로토콜에 대한 설명

경로 선택 metric 홉 카운트(hop count)이다

최단 경로 탐색에 Bellman-Ford 알고리즘을 사용한다

각 라우터는 이웃 라우터들로부터 수신한 정보를 이용하여 라우팅 표를 갱신한다

틀린 것: 라우팅 프로토콜을 IGP와 EGP로 분류했을 때 EGP에 해당한다

HRN 스케줄링: 최소 작업 우선(Shortest-Job-First) 기법의 약점을 보완한 비선점 스케줄링 기법

우선순위 = (대기한 시간 + 서비스를 받을 시간) / 서비스를 받을 시간

FIFO: First-In-First-Out

RR: Round-Robin, 프로세스들 사이의 우선순위가 아닌, 순서대로 일정 시간단위로 CPU를 할당

MQ: 5개의 프로세스 특성에 따라 각각 특정 큐에 할당하여 각 큐에서 독자적인 스케줄링, 큐마다 우선순위

UNIX 운영체제에 관한 특징

하나 이상의 작업에 대하여 백그라운드에서 수행이 가능하다

트리 구조의 파일 시스템을 갖는다

이식성이 높으며 장치간의 호환성이 높다

틀린 것: Multi-User는 지원하지만 Multi-Tasking은 지원하지 않는다

UDP 프로토콜의 특징

비연결형 서비스를 제공

단순한 헤더 구조로 오버헤드가 적다

TCP와 같이 트랜스포트 계층에 존재한다

틀린 것: 주로 주소를 지정하고, 경로를 설정하는 기능을 한다

Python 데이터 타입 중 시퀀스(Sequence) 데이터 타입에 해당, 다양한 데이터타입들을 주어진 순서에 따라 저장할 수 있으나 저장된 내용을 변경할 수 없는 것

튜플(tuple) 타입

응집도: 모듈의 독립성, 모듈 내부 구성요소 간 연관 정도

우연적 < 논리적 < 시간적 < 절차적 < 통신적 < 순차적 < 기능적

우연적(Coincidental): 서로 간에 어떠한 의미 있는 연관 관계도 없는 기능 요소로 구성될 경우

논리적(Logical): 유사한 성격을 갖거나 특정 형태로 분류되는 처리 요소들이 한 모듈에서 처리되는 경우

시간적(Temporal): 연관된 기능이라기 보다는 특정 시간에 처리되어야 하는 활동들을 한 모듈에서 처리할 경우

절차적(Procedural): 모듈이 다수의 관련 기능을 가질 때 모듈 안의 구성요소들이 그 기능을 순차적으로 수행할 경우

통신적(Communication): 동일한 입력과 출력을 사용하여 다른 기능을 수행하는 활동들이 모여 있을 경우

순차적(Sequential): 모듈 내에서 한 활동으로부터 나온 출력 값을 다른 활동이 사용할 경우

기능적(Functional): 모듈 내부의 모든 기능이 단일한 목적을 위해 수행되는 경우

IP 프로토콜의 주요 특징

패킷을 분할, 병합하는 기능을 수행하기도 한다

비연결형 서비스를 제공

Best Effort 원칙에 따른 전송 기능을 제공

틀린 것: 체크섬(Checksum) 기능을 데이터 체크섬만 제공한다 (헤더 체크섬만 제공, 데이터 체크섬 제공하지 않음)

LRU: Least-Recently-Used, 가장 최근에 사용하지 않은 페이지를 교체

FIFO: First-In-First-Out, 가장 먼저 메모리에 올라온 페이지를 가장 먼저 내보냄, 교체

OPT: Optimal, 앞으로 가장 오랫동안 사용하지 않을 페이지를 교체, 프로세스가 앞으로 사용할 페이지를 미리 알아야함, 구현 불가능

LFU: Least-Frequently-Used, 참조횟수가 가장 적은 페이지를 교체, 여러 개 LRU처럼 교체

MFU: Most-Frequently-Used, 참조횟수가 가장 많은 페이지를 교체

사용자 수준에서 지원되는 스레드가 커널에서 지원되는 스레드에 비해 가지는 장점

커널 모드로의 전환 없이 스레드 교환이 가능하므로 오버헤드가 줄어든다

틀린 것:

한 프로세스가 운영체제를 호출할 때 전체 프로세스가 대기할 필요가 없으므로 시스템성능을 높일 수 있다

동시에 여러 스레드가 커널에 접근할 수 있으므로 여러 스레드가 시스템 호출을 동시에 사용할 수 있다

각 스레드를 개별적으로 관리할 수 있으므로 스레드의 독립적인 스케줄링이 가능하다

한 모듈이 다른 모듈의 내부 기능 및 그 내부 자료를 참조하는 경우의 결합도

결합도(Coupling): 모듈 내부가 아닌 외부의 모듈과의 연관도, 모듈 간의 상호의존성

오류가 발생했을 때 전파되어 다른 오류의 원인이 되는 파문 효과(Ripple Effect)를 최소화

인터페이스가 정확히 설정되어 있지 않을 경우, 불필요한 인터페이스가 나타나 모듈 사이의 의존도는 높아지고 결합도가 증가

다른 모듈과 데이터 교류가 필요한 경우 전역변수(Global Variable)보다는 매개변수(Parameter)를 사용하는 것이 결합도를 낮추는데 도움이 된다

자료 < 스탬프 < 제어 < 외부 < 공통 < 내용

자료(Data): 모듈 간의 인터페이스로 전달되는 파라미터를 통해서만 모듈 간의 상호작용이 일어나는 경우

스탬프(Stamp): 모듈 간의 인터페이스로 배열이나 객체, 구조 등이 전달되는 경우

제어(Control): 모듈이 다른 모듈의 내부 논리 조직을 제어하기 위한 목적으로 제어 신호를 이용하여 통신하는 경우

외부(External): 외부 모듈이 다수의 관련 기능을 가질 때 모듈의 구성요소들이 그 기능을 순차적으로 수행할 경우

공통(Common): 파라미터가 아닌 모듈 밖에서 선언되어 있는 전역변수를 참조하고 전역변수를 갱신하는 식으로 상호작용하는 경우

내용(Content): 다른 모듈 내부에 있는 변수나 기능을 다른 모듈에서 사용하는 경우

모듈화와 관련한 설명

시스템을 모듈로 분할하면 각각의 모듈을 별개로 만들고 수정할 수 있기 때문에 좋은 구조가 된다

모듈 간의 결합도가 약해야 독립적인 모듈이 될 수 있다

모듈 내 구성 요소들 간의 응집도가 강해야 좋은 모듈 설계이다

소프트웨어의 모듈은 프로그래밍 언어에서 Subroutine, Function 등으로 표현될 수 있다

모듈화는 시스템을 지능적으로 관리할 수 있도록 해주며, 복잡도 문제를 해결하는데 도움을 준다

모듈화는 시스템의 유지보수와 수정을 용이하게 한다

틀린 것:

응집도는 모듈과 모듈 사이의 상호의존 또는 연관 정도를 의미한다

모듈 수 증가 - 상대적으로 각 모듈 크기 커짐 - 모듈 사이의 상호교류가 감소, 과부하(Overload) 현상이 나타난다

IP 주소체계와 관련한 설명

IPv6는 주소 자동 설정(Auto Configuration) 기능을 통해 손쉽게 이용자의 단말을 네트워크에 접속시킬 수 있다

IPv4는 호스트 주소를 자동으로 설정하며 유니캐스트(Unicast)를 지원한다

IPv4는 클래스별로 네트워크와 호스트 주소의 길이가 다르다

틀린 것: IPv6의 패킷 헤더는 32 octet의 고정된 길이를 가진다

IPv4:

32비트 주소

유니캐스트 / 멀티캐스트 / 브로드캐스트 사용

IPv6:

128비트 주소

기존 IPv4의 주소 부족 문제를 해결하기 위해 개발

인증성 / 기밀성 / 무결성 지원(=보안성 강화)

유니캐스트 / 애니캐스트 / 멀티캐스트 사용

C 언어 2차원 배열

arr [2][3]

(\*p)[3] = NULL

\*(p[0]) = arr[0]

\*(p[1]) = arr[1]

\*(p[0] + 0) = arr[0][0]

\*(p[1] + 1) = arr[1][1]

\*(p[1] + 2) = arr[1][2]

OSI 7계층

1계층 물리 계층(Physical Layer): 전기적, 기계적, 기능적인 특성을 이용해서 통신 케이블로 데이터를 전송, 비트 단위

통신케이블, 리피터, 허브

2계층 데이터링크 계층(DataLink Layer): 물리 계층을 통해 송수신되는 정보의 오류와 흐름을 관리, 안전한 정보의 전달을 수행할 수 있도록 도와주는 역할, 에러 검출, 재전송, 흐름제어, 맥주소를 가지고 통신, 프레임 단위

브리지, 스위치

3계층 네트워크 계층(Network Layer): 데이터를 목적지까지 가장 안전하고 빠르게 전달하는 기능(라우팅), 경로 선택, 주소를 정하고 경로에 따라 패킷을 전달

4계층 전송 계층(Transport Layer): 통신을 활성화하기 위한 계층, TCP프로토콜 이용, 포트를 열어 응용프로그램들이 전송, 패킷 생성 및 전송

5계층 세션 계층(Session Layer): 데이터가 통신하기 위한 논리적인 연결, TCP/IP 세션을 만들고 없애는 책임

6계층 표현 계층(Presentation Layer): 데이터 표현이 상이한 응용 프로세서의 독립성을 제공, 사용자의 명령어를 완성 및 결과 표현, 포장, 압축, 암호화

7계층 응용 계층(Application Layer): 최종목적지 HTTP, SMTP, POP3, IMAP, Telnet 등 프로토콜, 응용서비스를 수행, 네트워크 소프트웨어 UI부분, 사용자의 입출력 부분

패킷을 발신지로부터 최종 목적지까지 전달하는 책임을 진다

패킷에 발신지와 목적지의 논리주소를 추가한다

라우터 또는 교환기는 패킷 전달을 위해 경로를 지정하거나 교환 기능을 제공한다

틀린 것: 한 노드로부터 다른 노드로 프레임을 전송하는 책임을 진다(데이터링크 계층)

IPv6에 대한 특성

2^128개의 주소를 표현

등급별, 서비스별로 패킷을 구분할 수 있어 품질보장이 용이

확장기능을 통해 보안기능을 제공

틀린 것: 표시방법은 8비트씩 4부분의 10진수로 표시(16비트씩 8부분의 16진수로 표시)

JAVA의 Exception(예외)과 관련한 설명

오동작이나 결과에 악영향을 미칠 수 있는 실행 시간 동안에 발생한 오류

배열의 인덱스가 그 범위를 넘어서는 경우 발생하는 오류

존재하지 않는 파일을 읽으려고 하는 경우에 발생하는 오류

틀린 것: 문법 오류로 인해 발생하는 것(Syntax error, 컴파일 시 에러가 발생하므로 예외조건에 부합하지 않음)

ICMP(Internet Control Message Protocol): TCP/IP 계층 구조에서 IP의 동작 과정에서의 전송 오류가 발생하는 경우에 대비해 오류 정보를 전송하는 목적으로 사용하는 프로토콜

ARP(Address Resolution Protocol): IP 네트워크 상에서 IP주소를 MAC주소로 변환하는 프로토콜

PPP(Point-to-Point Protocol): 데이터링크 계층, 서로 다른 업체의 원격 액세스 소프트웨어들이 시리얼라인 상으로 연결하여 TCP/IP 프로토콜로 통신할 수 있도록 제정된 표준 규약

ECP(Error Checking Protocol): 데이터링크 계층 or 전송 계층

좋은 소프트웨어 설계를 위한 소프트웨어의 모듈간 결합도와 모듈 내 요소 간 응집도에 대한 설명

응집도는 높게, 결합도는 낮게 설계

Semaphore(세마포어): 임계 구역의 접근을 제어하는 상호배제 기법, 공유된 자원의 데이터 혹은 임계 영역 등에 따라 여러 프로세스 또는 스레드가 접근하는 것을 막아줌, 동기화 대상이 하나 이상

wait():

P(S): while S <= 0 do skip;

S := S – 1;

Signal():

V(S): S := S + 1;

Dekker Algorithm: 프로세스가 두 개일 때 상호 배제를 보장하는 최초의 알고리즘, flag와 turn 변수를 사용

Lamport Algorithm: 프로세스 n개의 상호 배제 문제를 해결한 알고리즘, 프로세스에게 고유한 번호를 부여하고 번호를 기준으로 우선순위를 정하여 우선순위가 높은 프로세스가 먼저 임계구역에 진입하도록 구현

Peterson Algorithm: 프로세스가 두 개일 때 상호 배제를 보장, 데커와 유사, 상대에게 진입기회를 양보, 더 간단

소프트웨어 개발에서 모듈이 되기 위한 주요 특징

다른 것들과 구별될 수 있는 독립적인 기능을 가진 단위(Unit)이다

독립적인 컴파일이 가능하다

유일한(Unique) 이름을 가져야 한다

모듈, 다른 프로그램에서 또 다른 모듈을 호출 가능

틀린 것: 다른 모듈에서의 접근이 불가능해야 한다

빈 기억공간의 크기가 20KB, 16KB, 8KB, 40KB일 때 기억장치 배치 전략으로 ‘Best Fit’을 사용하여 17KB의 프로그램을 적재할 경우 내부단편화(프로그램을 적재하고 나서 생기는 빈 공간)의 크기

20 – 17 = 3

16 – 17 = 불가

8 – 17 = 불가

40 – 17 = 23

3KB가 가장 작다

Best Fit: 가능한 공간 중 가장 작은 기억공간을 사용

First Fit: 가능한 공간 중 가장 첫번째 기억공간을 사용

Worst Fit: 가능한 공간 중 가장 큰 기억공간을 사용

개발 환경 구성을 위한 빌드(Build) 도구

Build: 작성된 소스 코드를 실제 기기에서 실행할 수 있는 소프트웨어로 변환하기 위한 과정

Ant: 아파치 재단에서 개발한 자바의 공식적인 빌드 도구

Maven: 아파치 재단에서 Ant 대안으로 개발한 빌드 도구

Gradle: Ant, Maven의 보완으로 개발된 빌드 도구, 안드로이드 스튜디오 주 빌드 도구

틀린 것: Kerberos; 컴퓨터 네트워크 인증 암호화 프로토콜

연산자 우선순위

1순위: ++(후위), --(후위), (), [], .(참조선택), ->(포인터)

2순위: !, ~, +, -, ++(전위), --(전위), (타입), \*(참조), &(주소), sizeof

3순위: \*(곱셈), /, %

4순위: +, -

5순위: <<, >>

6순위: <, <=, >, >=

7순위: ==, !=

8순위: &(비트 and)

9순위: ^(XOR)

10순위: |

11순위: &&(논리 and)

12순위: ||

13순위: ?:(삼항 조건 연산자)

14순위: 대입 연산자 및 복합 대입 연산자(+=, -=, etc)

15순위: ,

a=1, b=2

a < b + 2 && a << 1 <= b, (b+2)

a < 4 && a << 1 <= b, (a<<1)

a < 4 && 2 <= b, (a<4), (2<=b)

1 && 1

1

UNIX 시스템의 Shell의 주요 기능에 대한 설명

사용자 명령을 해석하고 커널로 전달하는 기능 제공

반복적인 명령 프로그램을 만드는 프로그래밍 기능을 제공

초기화 파일을 이용해 사용자 환경을 설정하는 기능을 제공

틀린 것: 쉘 프로그램 실행을 위해 프로세스와 메모리를 관리한다(프로세스, 메모리 관리는 커널의 기능)

오류 제어에 사용되는 자동 반복 요청방식(Automatic Repeat request, ARQ)

Stop-and-Wait ARQ: 송신 측이 하나의 블록을 전송한 후 수신 측에서 에러 발생을 점검, 에러발생 유무 신호 기다림

Go-Back-N ARQ: 여러 블록을 연속적으로 전송하고, 수신 측에서 부정 응답(NAK)을 보내오면 송신 측이 오류가 발생한 블록부터 모두 재전송

Selective-Repeat ARQ: 여러 블록을 연속적으로 전송하고, 수신 측에서 부정 응답(NAK)을 보내오면 송신 측이 오류가 발생한 블록만을 재전송

Adaptive ARQ: 전송 효율을 최대로 하기 위해서 데이터 블록의 길이를 채널의 상태에 따라 동적으로 변경하는 방식

Garbage Collector: Java에서 힙(Heap)에 남아있으나 변수가 가지고 있던 참조값을 잃거나 변수 자체가 없어짐으로써 더 이상 사용되지 않는 객체를 제거해주는 역할을 하는 모듈

Java 연산자 우선순위

-- > % > & > =

SSTF: Shortest Seek Time First, 현재 헤드 위치에서 가장 가까운 것부터,

가장 가까운 것 > 가장 이동 거리가 짧은 것 >= 현재 트랙과 남아있는 트랙 중 차이가 적은 것

CIDR 표기법 IPv4 32비트(255.255.255.255) 중 앞 24비트가 네트워크 IP, 뒤 8비트가 호스트 IP,

4개의 서브넷으로 나누려면 호스트 IP의 앞쪽 비트를 가능한 적게 4개까지 나누어야 함

4개로 나누려면 2비트를 사용해야하기 때문에 앞 2비트를 네트워크 IP로 넘겨줌 192.168.1.0/24 -> 192.168.1.0/26

192.168.1.(00) // 00 0000 (0)

192.168.1.(01) // 00 0000 (64)

192.168.1.(10) // 00 0000 (128)

192.168.1.(11) // 00 0000 (192) 로 나눌 수 있음

IP subnet-zero를 적용: 호스트 IP가 전부 0인 192.168.1.0도 사용하겠다는 것을 의미

단 각 서브넷에서 호스트IP가 전부 0인 주소와 전부 1인 주소는 다른 용도로 이미 예약, 0; 네트워크 자체 주소, 1;브로드캐스팅 주소

따라서 192.168.1.11 // 00 0000(192)은 빼고 192.168.1.11 // 00 0001(193)부터 IP 사용 가능

IP Address: A Class, B Class, C Class

A Class: 0.0.0.0 ~ 127.255.255.255

B Class: 128.0.0.0 ~ 191.255.255.255

C Class: 192.0.0.0 ~ 223.255.255.255

D Class: 224.0.0.0 ~ 239.255.255.255, 멀티캐스트용

E Class: 240.0.0.0 ~ 225.255.255.255 연구용

프로세스와 관련한 설명

프로세스가 준비 상태에서 프로세서가 배당되어 실행 상태로 변화하는 것을 디스패치(Dispatch)라고 한다

프로세스 제어 블록(Process Control Block, PCB)은 프로세스 식별자, 프로세스 상태 등의 정보로 구성

이전 프로세스의 상태 레지스터 내용을 보관하고 다른 프로세스의 레지스터를 적재하는 과정을 문맥 교환(Context Switching)이라고 한다

틀린 것: 프로세스는 스레드 내에서 실행되는 흐름의 단위이며, 스레드와 달리 주소 공간에 실행 스택(Stack)이 없다

디스패치(Dispatch): 준비 -> 실행

할당시간초과(Time Run Out): 실행 -> 준비

대기(Block): 실행 -> 대기

깨움(Wake Up): 대기 -> 준비

PCB: 운영체제가 그 프로세스를 관리하는데 필요한 모든 정보를 유지하는 자료구조 테이블

* 상태, 소유자, 실시간 통계, 스레드, 관련 프로세스 리스트, 자식 프로세스 리스트, 주소공간, 자원, 스택

Context Switching: CPU가 현재 실행하고 있는 프로세스의 상태를 PCB에 저장하고 다음 프로세스의 PCB로부터 문맥을 복원하는 작업

프로세스의 실행 단위를 스레드라고 부른다

TCP 헤더와 관련한 설명

순서번호(Sequence Number)는 전달하는 바이트마다 번호가 부여된다

수신번호확인(Acknowledgement Number)은 상대편 호스트에서 받으려는 바이트의 번호를 정의

체크섬(Checksum)은 데이터를 포함한 세그먼트의 오류를 검사

틀린 것: 윈도우 크기는 송수신 측의 버퍼 크기로 최대크기는 32767bit이다(16비트, 2^16 = 65535byte=64kb)

페이지 교체 알고리즘(Page Replacement): FIFO(First-In-First-Out), Optimal, LRU(Least Recently Used)

파일 디스크립터(File Descriptor)에 대한 설명

파일마다 따로 있으며 시스템에 따라 다른 구조를 가질 수 있다

파일 시스템에서 관리하므로 사용자는 직접 참조할 수 없다

파일 이름. 위치, 구조, 보조기억장치의 유형, (텍스트/이진/기계어/실행파일 등) 파일 유형, 시간, 액세스 등의 정보

파일 관리를 위해 시스템이 필요로 하는 정보를 가지고 있다

보조기억장치에 저장되어 있다가 파일이 개방(Open)되면 주기억장치로 이동된다

파일 제어 블록(File Control Block)이라고도 한다

틀린 것: 사용자가 파일 디스크립터를 직접 참고할 수 있다

CSMA/CA: 무선랜에서 데이터 전송 시, 매체가 비어 있음을 확인한 뒤 충돌을 회피하기 위해 임의 시간을 기다린 후 데이터를 전송하는 방법, 네트워크에 데이터의 전송이 없는 경우라도 동시 전송에 의한 충돌을 대비하여 확인 신호를 전송, 충돌 방지

Collision Domain: 충돌 발생을 검출할 수 있는 브리지 간 혹은 다른 계층 장치 간의 이더넷 세그먼트 범위

CSMA/CD: 이더넷에서 각 단말이 전송 공유 매체에 규칙있게 접근하기 위한 매체 액세스 제어 방식, 충돌 감지

교착 상태의 해결 방법

Detection(탐지): 교착 상태 발생을 허용하고 발생 시 원인을 규명하여 해결(ex 자원 할당 그래프)

Avoidance(회피): 교착 상태 가능성을 배제하지 않고 적절하게 피해나가는 방법(ex 은행원 알고리즘)

Recovery(복구): 교착 상태 발견 후 현황 대기를 배제시키거나 자원을 중단하는 메모리 할당 기법(ex 선점, 프로세스 중지; 희생자 선택)

Prevention(예방): 교착 상태의 필요조건을 부정함으로써 교착 상태가 발생하지 않도록 미리 예방하는 방법(ex 환형 대기, 비선점, 점유와 대기, 상호배제 4가지 부정)

CIDR(Classless Inter-Domain Routing)표기로 203.241.132.82/27, 서브넷 마스크

IPv4는 32bit, IP주소/서브넷마스크 로 표기 -> 1 이 27개

11111111.11111111.11111111.11100000 -> 255.255.255.224

프로세스 적재 정책과 관련한 설명

반복, 스택, 부프로그램은 시간 지역성(Temporal Locality)과 관련이 있다

공간 지역성(Spatial Locality)은 프로세스가 어떤 페이지를 참조했다면 이후 가상주소 공간 상 그 페이지와 인접한 페이지들을 참조할 가능성이 높음을 의미한다

스레싱(Thrashing) 현상을 방지하기 위해서는 각 프로세스가 필요로 하는 프레임을 제공할 수 있어야 한다

틀린 것: 일반적으로 페이지 교환에 보내는 시간보다 프로세스 수행에 보내는 시간이 더 크면 스레싱이 발생한다

스레싱: 일반적으로 페이지 수행 시간보다 교환시간이 커질 때 발생하는 것

프레임워크(Framework)에 대한 설명

소프트웨어 구성에 필요한 기본 구조를 제공함으로써 재사용이 가능하게 해준다

특징: 모듈화, 재사용성, 확장성, 제어의 역흐름

틀린 것:

소프트웨어 개발 시 구조가 잡혀 있기 때문에 확장이 불가능하다

소프트웨어 아키텍쳐와 동일한 개념이다

모듈화가 불가능하다

리눅스 Bash Shell에서 export와 관련한 설명

export가 매개변수 없이 쓰일 경우 현재 설정된 환경변수들이 출력된다

사용자가 생성하는 변수는 export 명령어 표시하지 않는 한 현재 쉘에 국한된다

변수를 export 시키면 전역(global) 변수처럼 되어 끝까지 기억된다

틀린 것: 변수를 출력하고자 할 때는 export를 사용해야 한다(echo $변수명)

페이징 기법에서 페이지 크기가 작아질수록 발생하는 현상

페이징 기법: 컴퓨터가 메인 메모리에서 사용하기 위해 데이터를 저장하고 검색하는 메모리 관리 기법

페이징 기법을 통해 물리적 메모리는 연속적으로 할당되어 존재할 필요가 없음, 비 연속적 메모리를 연속적 메모리처럼 만들 수 있음

더 많은 페이징 사상 테이블 필요

내부 단편화 감소

페이지의 집합을 효율적으로 운영 가능

기억장치의 효율이 좋음

총 입출력 시간 증가

페이지 크기가 큰 경우

주 기억장치 공간 절약

참조되는 정보와 무관한 양의 정보가 주 기억장치에 남게 됨

테이블이 복잡하지 않아 관리 용이

TCP 프로토콜과 관련한 설명

흐름 제어(Flow Control)의 기능을 수행

전이 중(Full Duplex) 방식의 양방향 가상 회선을 제공한다

전송 데이터와 응답 데이터를 함께 전송할 수 있다

패킷의 전송 및 오류를 제어한다

틀린 것: 인접한 노드 사이의 프레임 전송 및 오류를 제어한다(데이터 링크 계층에 해당하는 설명)

C 언어 라이브러리 중 stdlib.h에 대한 설명

문자열을 수치 데이터로 바꾸는 문자 변환 함수와 수치를 문자열로 바꿔주는 변환 함수 등이 있다

문자열 변환, 사 난수 생성, 동적 메모리 관리 등 함수 포함

틀린 것:

문자열 처리 함수로 strlen()이 포함되어 있다(string.h 포함)

표준 입출력 라이브러리이다(stdio.h)

삼각함수, 제곱근, 지수 등 수학적인 함수를 내장하고 있다(math.h)