1. 산업혁명

1차산업혁명 : 기계, 증기기관, 이동의 혁명, 물류 유통을 통한 산업 혁명의 파급력 배가

2차산업혁명 : 1차 세계대전 직전, 전기 분야, 컨베이어 벨트 시스템 -> 대량 생산

3차산업혁명 : 기계 -> 디지털 기술로의 발전, 인터넷 및 정보통신기술(ICT)의 발전으로 자동화, 정보화 시대, 다양한 통신 매체

4차산업혁명 : 가상~물리 세계가 서로 결합되는 사이버 물리 시스템, 스마트 팩토리, 빅데이터 IoT, 인공지능, 모바일, 3D프린터, 무인자동차, 클라우드 컴퓨팅, 가상현실

2. 소프트웨어 중심사회

SW가 혁신과 성장, 가치 창출의 중심이 되고 개인, 기업, 국가의 경쟁력을 좌우함

소프트웨어 교육 : 주요 정책
(1) 청소년들의 sw 교육기회 확대
(2) 모든 대학에 실전적 소프트웨어 교육이 대폭 확대
(3) 정부는 SW 기반의 새로운 미래 성장 동력 창출을 지원
(4) SW로 제조업의 고부가가치화 촉진
이점 : 문제해결 과정에서의 사고력 증진

3. 컴퓨터의 발전

(기계식 - 전기 기계식 – 전자식)

기계식 : 주판. 네이피어 막대. 파스칼의 계산기. 라이프니치의 계산기. 배비지의 해석기관

전기 기계식 : 홀러리스의 천공카드 시스템. 튜링기계. MARK-I

전자식 컴퓨터 : ABC(최초의 전자식 컴퓨터). ENIAC(최초의 범용 전자식 컴퓨터)

4. 전자식 컴퓨터의 발전 과정

- **1세대 (진공관을 기억소자로 사용)**

에니악(최초의 전자식 컴퓨터). 에드삭(최초의 프로그램 내장 방식). 에드박(이진수 사용 컴퓨터). 유니박(최초의 상업용)

- **2세대 (트랜지스터를 기억소자로 사용)**

트랜지스터 사용 -> 컴퓨터의 크기 대폭 감소.
포트란, 코볼 이라는 고급 수준의 프로그래밍 언어가 개발됨.&구조적 프로그래밍 기법 등장

- **3세대 (집적회로(IC)가 기억소자로 사용됨)**

IBM S/360이 상용화됨. 소프트웨어의 체계 확립 시기
입력한 작업을 시간을 나누어 수행
-> 시분할 운영체제 사용
컴퓨터가 동시에 여러 작업을 수행할 수 있는 다중프로그래밍 기술 적용
C, Unix 등장. 에디터, 컴파일러, 인터프리터, 디버거 등의 도구가 하나로 합쳐진 IDE 방식 사용

- **4세대(고밀도 및 초고밀도 집적회로(LSI, VLSI)를 기억소자로 사용)**

마이크로프로세서 : 메모리 및 연산, 논리, 제어 회로등을 하나의 고밀도 집적회로 칩에 담음
마이크로소프트 설립. 16비트 도스 방식
-> 32비트 윈도우 방식으로 변화. JAVA 개발됨.
넷스케이프, 익스플로러 등의 웹브라우저 등장.

5. 컴퓨터의 개념 및 구성

컴퓨터 : 전자 회로를 이용한 고속의 자동 계산기

- **폰 노이만 구조**

- CPU 처음 고안. 에드삭 개념의 시초
- CPU, 메모리, 입력 장치, 출력 장치로 구성된 구조.
- 프로그램 내장방식

(프로그램과 데이터 모두 메인 메모리에 저장됨, CPU는 메인 메모리에 저장된 프로그램에서 명령어들을 순차적으로 가져와 수행)

- CPU : 산술/논리장치(직접 연산 수행)와 제어장치(CPU와 다른 장치들 사이의 데이터 흐름 제어)로 구성됨

장점 : 하드웨어 재배치 없이 소프트웨어만 교체하면 됨(범용성)

단점 : 내장 메모리 순차처리 방식. 데이터 메모리와 프로그램 메모리가 구분되지 않고. 하나의 버스를 가지고 있는 구조
CPU가 명령어와 데이터에 동시 접근 못함.(병목현상)

• **컴퓨터의 구조**

1) 중앙처리장치 (제어장치, 산술/논리장치, 레지스터)

컴퓨터 시스템 제어. 프로그램 명령 실행.
주기억장치에 저장되어 있는 명령어를 가져와 실행.
명령어는 0과 1로 이루어진 이진 패턴.
다른 장치들과는 시스템 버스로 연결되어 있음.

- 제어 장치 : 프로그램 명령어 해석. 연산장치, 주기억장치, 입출력 장치 등에게 동작을 지시
- 연산장치 : 사칙연산 & 논리 연산(AND, OR, NOT, XOR) 수행
- 레지스터 : 주기억장치로부터 읽어온 명령어나 데이터를 저장하거나 연산된 결과를 저장하는 공간

2) 주기억장치(RAM, ROM)

- RAM : 휘발성(전원이 꺼짐년 데이터가 지워짐)
- ROM : 읽기만 가능한 저장매체. 비휘발성. BIOS (컴퓨터가 부팅되면서 설정값들을 읽어오는 장치)존재.

+) 입출력장치(키보드, 마우스, 웹캡, 모니터, 프린터), 보조기억장치(HDD, SSD, DVD)

6. 하드웨어와 소프트웨어

• **하드웨어**

물리적인 장치 (CPU, 주기억장치, 보조기억 장치, 입력장치, 출력장치)

- CPU : 시스템에 부착된 모든 장치의 동작 제어& 명령 실행(제어장치)//& 자료의 연산 수행(연산장치)& 연산에 필요한 자료를 임시로 저장(레지스터)
- 버스 : 여러 장치들을 연결하는 공유 전송 매체. 데이터와 각종 제어신호 전달
- 메인보드(마더보드) : 중앙처리장치를 연결하는 소켓 등 다른 하드웨어 장치에 연결시키는 확장슬롯, ROM으로 구성
- 주기억장치 : 각 전자회로에 비트 단위의 데이터가 저장됨.
- 보조기억장치 : 프로그램이나 데이터를 저장하기 위한 저장장치 / 비휘발성
- 입출력장치 : 키보드, 마우스.. 모니터, 스피커, 프린터

• **소프트웨어**

- **시스템 소프트웨어**

컴퓨터 운영에 필요한 프로그램.(하드웨어, 운영체제, 응용 소프트웨어 관리)
Windows, Mac OS, Unix, Linux
컴파일러, 인터프리터, 어셈블러
백신, 보안, 압축, 디스크관리 프로그램, DBMS

- **응용 소프트웨어**

한글, 엑셀, 파워포인트..
오토캐드, 포토샵, 일러스트, Visual Studio, Python 개발도구 등.

7. 데이터의 표현

저장 용량 : B - KB - MB - GB - TB - PB – EB

데이터 저장 용량 비교 : 비트(0, 1 / 예 아니오) - 니블(4비트) - 바이트(8비트) - KB - MB - GB – TB

데이터의 표현
2진수 : 1101(2) = 1 + 4 + 8 = 13
16진수 : 0~9 , A ~ F

8. 컴퓨팅 사고

<지넷 윙>에 의해 최초 언급

• **지넷윙의 컴퓨팅 사고 5요소**

분해 - 개념화 - 추상화 - 재귀적 사고(해답을 찾은 후 해법을 문제 해결에 지속적이고 반복적으로 적용할 수 있는 사고) - 병렬 처리

누구에게나 일반적으로 적용되는 사고방식
=>일반인에게도 필요하다
(컴퓨팅의 기본적인 개념과 원리를 기반으로 일상 생활에서 발생할 수 있는 여러 문제들을 효율적으로 해결할 수 있는 사고 능력)

효율적인 문제 해결
인간의 사고능력 / 컴퓨터의 능력을 통합한 것.
논리적& 창의적인 문제해결방식을 사람이 생각하고 컴퓨터에게 명령
S/W교육의 필요성
복합접 사고 -> 창의적 문제해결의 핵심 능력
복잡 어려운 애매한. 자신감 . 지속성. 인내심 / 해답에 대해서는 다양함(Opened)
공통적인 문제해결을 위한 다른 사람들과의 의사소통 능력(협업)
문제를 해결한 후 정확한 답이 나오는가./ 가장 효율적인 해결방법은/ 최소한의 자원 / 다른 문제에 적용이 가능한가(일반화/재사용성)

(1) 문제분석 : 논리적 분석. 점검. / 문제에 모순? 명확성. 열린사고문제형태?

(2) 데이터 수집/표현

(3) 분해 : 복잡한 문제를 작은 문제들로 분해하여 해결 (ex. 이진탐색(업다운))

(4) 패턴인식 : 유사성과 반복성을 가진 현상을 찾아내는 것. 실생활 패턴인식(수학 공식) 경영학, 경제학, 수열

(5) 추상화 : 필요없는 부분 제거. 중요한 핵심만 남겨둠. 요약단계 . 일반적인 원리를 찾아냄.(일반화, 핵심 특징 추출

(6) ★알고리즘 : 문제 해결을 위해 추상화된 핵심 원리를 일련의 공식 또는 절차로 나타내는 과정 / 문제 해결을 위한 논리적인 지시서

(7) 평가 : 해답의 적절성, 효율성 등을 최종점검(정확성, 효율성, 오류파악) -> 알고리즘을 기반으로 코딩

볼드체 4가지 요소를 모두 적용한것 : 복합적 컴퓨팅 사고, 볼드체는 상황에 따라 선택적으로 거치는 컴퓨팅 사고 핵심 요소

9. 소프트웨어 프로그래밍

컴퓨터에게 명령 : 이해할 수 있는 언어로

프로그래밍 언어 : 간결, 가독성, 명확성(문법에 따라 정확하게), 독립성(서로 다른 컴퓨터 상에서도 실행 가능)

최초의 고급언어 : 포트란// -> 코볼 -> (B)CPL, Algol, 등등.

핵심적인 내용만. 다른사람이 읽더라도 이해할 수 있게 가독성을 위해 & 누구나 읽기 편하게 - '주석'이라는 기능 사용

폰노이만 : 내장 프로그램 방식의 컴퓨터 구조 = 순차처리 구조 = 프로그램을 미리 메모리에 설치.저장

기계어 : 컴퓨터가 이해하기 쉽게 만들어진 언어(처음부터 이진코드로 작성)

어셈블리어 : 기계언어 중에서 특정 기호를 의미있는 단어로 지정.(add, sum, mov...) but, 다루는 데이터 자체는 이진표기를 사용
기계가 알아듣게 번역해주는 것 : 어셈블러

기계에 가까우면 저급언어 / 사람에 가까우면 고급언어 (빠른 개발을 위해 탄생) -----> 처리기준에 따른 분류

고급언어 : 사람이 이해하기 쉬운 방식, 하드웨어와 관련된 지식이 없어도 작성이 가능하다.
C언어는 하드웨어를 조금은 건들기는 했었음..

소스코드 : 프로그래밍 작성. / 작성된 명령어들의 집합
언어 번역 프로그램 : 컴파일러(최초의 방식), 인터프리터

- **컴파일러와 인터프리터**

컴파일러 : 소스코드를 한번에 번역 ->exe 파일생성 / 소스파일을 기계어로 번역하는 과정을 컴파일 이라고 함. 실행속도가 빠름 / 오류사항이 있으면 이를 수정하고 전체를 다시 컴파일

인터프리터 : 소스코드를 줄 단위로 바로 번역/ 실시간 통역사 / 실행파일은 생성X / 실행결과를 바로 확인 가능, 오류에 대한 빠른 피드백이 가능 / 줄단위이기 때문에 컴파일러보다 속도가 느림

- **절차지향 / 객체지향**

절차지향 : 순차적으로 작업 수행, 가독성이 좋음, 컴퓨터 처리방식과 유사 - 실행속도 빠름
절차적 프로그래밍 : 가장단순. 알고리즘에 의하여 단계별, 순차적 : 구조가 단순, 속도 빠름 , 코드 수정 및 유지보수 어려움

★객체지향 : 절차적이면서도 모듈화된 객체들을 어디서든 사용 가능하게 작업을 수행
절차적의 확정 / 객체를 생성하여 객체를 조립해가며 완성. 재사용성

캡슐화(기능처리 함수를 묶어 재사용),
정보은닉(외부로부터 정보를 숨김, 접근 제어),
다형성(하나의 이름으로 여러 기능을 수행),
상속(상위 클래스의 기능을 재사용),
추상화(객체의 속성을 모델링) : 소스코드 재사용성, 대형프로젝트에 적합 / 복잡한 구조, 설계 및 저장 공간 비용 높음

- **프로그래밍 언어 예시**

- C언어 : 간단한 설계, 프로그래밍 언어의 기본, 신뢰성,규칙성,간소성, 저급언어의 기능 구형, 코드가 간결하고 실행 빠름

융통성, 이식성,풍부한 연산자.등등

- C++ : 게임이나 그래픽 분야. C언어의 유연성에 객체지향의 편리성 접목

- C# : MS에서 제작. 윈도우에서 동작하는 대부분의 프로그램에 사용 / C보다는 Java에 가까운 객체지향 언어

- Java : 만들어진 프로그램이 JVM(Java Virtual Machine)에서 실행 / 컴파일러에 의해 바이트코드 파일로 변환, 인터프리터 방식으로 실행.
  SW / OS(os / jvm) / HW

- 파이썬 : 객체지향이면서 플랫폼에 독립적, 인터프리터 언어 / 웹사이트, 서비스 개발, 데이터 분석 등등
  파이썬 (+ R) : 데이터분석, 통계

- 블록코딩 : 스크래치

10. 알고리즘

어떠한 일을 논리적인 절차와 명령어들로 이루어진 집합 / 문제해결 과정 - 컴퓨터 코딩의 기초 과정
오류사항, 예외사항이 발생했을 때 중요함
내가 해결하고자 하는 문제에 적합/부합한지 & 처리속도에 문제가 없는지 & 문제해결의 정확도
의사코드와 순서도 작성

- **알고리즘의 조건 5가지**

- input : 없을 수도 있음
- output : 1개 이상의 결과를 가짐
- 명확성 : 모호하지 않은
- 유한성 : 실행이 유한한 횟수(무한반복x)
- 유효성 : 실행이 가능한 것이어야.

동일한 문제 유형에 일반적으로 적용하여 해결이 가능한가 -> **일반화**
다른사람이 보기에 이해하기 쉬운 알고리즘인가
최소한의 자원과 시간을 사용하는 알고리즘 구성

시간복잡도(걸리는 시간)와 공간복잡도(메모리공간사용)

- **의사코드**

가상의 코드 != 실제코드 but 프로그래밍언어와 유사함

```
begin
a = 1
b = 2
print a + b
while 조건문
        실행문
END while
end
```

a와 b의 값을 서로 바꾸고 싶으면 임시 저장소를 생성해서 값을 하나 저장한 다음 대입 / 대입

- **순서도**

도형 및 기호	이름	설명
	단말	순서도의 시작과 끝을 나타냄
	준비	작업단계 시작 전 준비 (변수 및 초기화 단계)
	처리	처리작업 명시
	입출력	입출력 내용 기재
	비교/ 판단	여러 조건에 대한 흐름선 구분
	흐름선	프로그램의 흐름