**객체지향프로그래밍 : 2차시**

**기계어**0과 1로 구성된 이진수 언어. 컴퓨터의 CPU는 기계어만 처리.

**어셈블리어**기계어의 명령을 ADD, SUB, MOVE 등과 같은 상징적인 니모닉 기호로 일대일 대응시킨 언어.

* 니모닉 기호 : 기계어로 구성된 일련의 숫자를 알아보기 쉽게 언어(단어)로 바꾸어 준 것.  
  (ex. 00000000 00000000 10110000 01100001 -> mov al. 061h)
* 어셈블러 : 어셈블리어 프로그램을 기계어 코드로 변환해 주는 일종의 번역 프로그램.

**고급언어**사람이 이해하기 쉬운, 복잡한 작업, 자료 구조 및 알고리즘을 표현하기 위해 고안된 언어.  
(ex. Pascal, Basic, C/C++, Java, C# . . .)

* 컴파일러 : 고급 언어로 작성된 프로그램을 기계어 코드로 변환해 주는 일종의 번역 프로그램.

텍스트, 스크린샷, 도표이(가) 표시된 사진

자동 생성된 설명

위 이미지처럼, 고급언어 -> 어셈블리어 -> 기계어 순으로 번역이 이루어짐.

**표준 C++**비 표준 C++ 프로그램 (ex. Visual C++, Borland C++ 등 컴파일러 회사 고유의 비 표준 구문)을 사용하면 특정 C++ 컴파일러에서만 컴파일이 가능하므로, 표준 C++ 프로그램을 통해 작성하여야 모든 플랫폼, 모든 표준 C++ 컴파일러에 의해 컴파일 될 수 있으므로 표준 C++컴파일러를 사용하여야 함.

텍스트, 도표, 스크린샷이(가) 표시된 사진

자동 생성된 설명

**C++의 설계 목적**

* 호환성
* 객체지향
* 타입체크
* 실행시간 효율성 최적화

**C언어에서부터 추가된 기능들**

* 디폴트 매개 변수(default parameter) :  
  매개 변수에 디폴트 값이 전달되도록 함수 선언

**C++의 객체지향 개념**소프트웨어의 재사용을 통해 생산성을 향상시키는 프로그래밍 방식.  
복잡하고 큰 규모의 소프트웨어에서의 작성 및 관리, 유지보수에 용이함.

* 캡슐화 :   
  데이터를 캡슐로 싸듯이 클래스 등으로 캡슐화하여 외부의 접근으로부터 보호 및 분리함.  
  클래스 – 객체를 만드는 틀  
  객체 – 클래스라는 틀에서 생겨난 실체  
  객체와 인스턴스는 같은?뜻?? 미묘하게 다르긴 한데 이 강의에서는 같다고 하는듯.
* 상속성 :   
  자식 클래스나 메서드 등이 부모 클래스나 메서드의 선언된 모양 그대로의 멤버들을 가지고 탄생하는 것.  
  텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

  자동 생성된 설명
* 다형성 :   
  하나의 기능을 여러 경우에 따라 다르게 작동할 수 있게 설정하는 것.  
  연산자 중복, 함수 중복, 함수 재정의(오버라이딩) 등이 존재.  
  연산자 중복 – ‘+’, ‘-‘, ‘\*’ 등의 연산자가 다양한 자료형에 중복되어 적용될 수 있음.  
  함수 중복 – 함수 내의 매개변수 조건을 여러가지로 설정하여, 다른 조건 내에서도 사용할 수 있도록 설정.  
  함수 재정의 – 덮어씌우듯이, 부모 함수의 값을 상속받아 재정의함.

**제네릭 함수**동일한 프로그램 코드에 다양한 데이터 타입을 적용할 수 있게 일반화 시킨 함수.  
템플릿 쓰는게 제네릭

**C++의 호환성 추구**  
C++은 C언어와의 호환성을 추구하므로, C의 코드 및 헤더를 사용할 수 있음.  
단, 섞어 쓸 경우 캡슐화가 무너지기 쉬움.(C는 절차지향 언어이기 때문.)