

I - 1. 정보과학과 우리 삶은 어떤 관련이 있을까

1. 정보 사회와 정보 과학

정보 사회 : 다양한 자료의 수집, 분석, 처리 및 공유 등을 통한 정보의 생산과 활용이 모든 가치 판단의 중심이 되는 사회.

→ 정보과학의 지식과 기술을 활용해 일상 생활의 여러가지 문제를 효율적으로 해결함.

→ 정보 사회는 소프트웨어 중심 사회로 진입하고 있는 사회라 할 수 있음.

정보 과학 : 일상생활의 문제를 효율적으로 해결하기 위해 컴퓨팅 시스템의 계산 능력을 이용하는 방법이나 지식을 연구하는 학문

→ 정보 기술 : 정보 과학으로 연구되어 일상생활의 문제를 해결하기 위해 사용되는 방법이나 지식

2. 정보과학 지식과 기술의 활용 분야

O2O	온라인과 오프라인이 결합하는 것. 즉, 온라인에서 고객 유지 → 오프라인에서 상품 구매 유도
핀테크	모바일 기기를 이용해 경제 활동을 할 수 있는 금융과 정보 기술이 융합된 산업
드론	촬영, 범죄 수사, 물류 등 다양한 영역에서 활동할 수 있는 초소형 비행체
자율 주행 자동차	첨단 컴퓨터·통신·측정 기술을 장착하고 있어 사람이 없어도 직접 도로 주행을 할 수 있음
디지털 사이니지	네트워크를 통한 원격제어가 가능한 디지털 디스플레이를 설치해 정보·광고 등 미디어 제공
헬스 케어	신체에 부착된 기기가 개인의 건강 기능을 체크하여 언제 어디서나 건강 관리를 받을 수 있음
무크	누구나, 언제 어디서나 들을 수 있는 무료 온라인 공개강좌 서비스
실감 미디어	HMD*기기와 가상 현실 등으로 실감나는 콘텐츠를 제공하는 것

*HMD : 안경처럼 머리에 쓰고 대형 영상을 즐길 수 있는 영상 표시 장치 (ex. VR 헤드셋)

3. 정보과학과 미래 직업

소프트웨어 개발자	다양한 소프트웨어 개발 작업(프로그램 설계, 코딩, 프로젝트 관리 등)에 참여하는 사람
데이터 과학자	대량의 자료를 분석하고 관리하여 사람들의 행동 패턴이나 시장 경제 상황을 예측하는 사람
로봇 공학자	로봇의 구조와 동작 원리를 이해하여 실생활에서 사용할 수 있는 로봇을 개발하는 사람
네트워크 관리자	네트워크와 관련된 각종 하드웨어 및 소프트웨어에 관한 업무를 수행하는 사람
가상 현실 전문가	컴퓨터 그래픽을 이용해 가상의 3차원 시공간을 구상하고 설계하는 사람
사물 인터넷 개발자	사물에 센서를 부착하여 실시간으로 데이터를 주고받는 기술을 개발하고 응용하는 사람

I - 2. 정보 공유와 정보 보호는 어떻게 실천할까

1. 정보 공유

정보 공유 : 정보를 다른 사람과 나누는 것. 정보 공유는 공통 관심사를 가진 사람들을 하나로 묶어 새로운 인간관계를 만들기도 하고, 협업이나 새로운 아이디어를 창출하여 여러 가지 형태의 창업을 가능하게 하는 원동력이 됨.

정보 공유의 가치 :

공유	인터넷을 통한 자료 및 정보의 공유, 인터넷 강의 등
협업	화상 회의를 통한 협업, 온라인 메모, 온라인 일정 관리 등
소통	블로그, 메신저, 밴드 활동 등을 통한 의사소통, 친구 모임, 동호회 활동 등
개방	폭염 주의보 문자 전송과 같은 공공 기관에서의 정보 공시, 정부 및 공공기관에서의 자료 공시 등

2. 정보 보호와 관련 제도

정보 보호 : 정보가 불법적으로 유출, 훼손, 파괴, 변조, 접근, 기록되는 것으로부터 보호 등을 하기 위한 모든 행위를 말함.

→ 정보 사회의 구성원이 올바른 정보 공유를 통해 다양한 가치를 얻을 수 있게 함.

보호해야 할 정보의 예 :

군사 정보	군사 비밀 정보, 군사력 정보, 군부대 위치 등
금융 정보	계좌번호, 대출 정보, 카드 결제 내역 등
위치 정보	사람이나 자동차의 이동 경로, 거주지 및 건물의 위치 정보, 주차 위치 정보 등
인적 정보	주민 등록 번호, 휴대전화 번호, 집 주소 등
의료 정보	처방전, 수술 이력, 진료 기록 등
영상 정보	방범 CCTV, 블랙박스, 과속 감시 카메라 등

정보 유출로 악의적인 목적으로 정보가 사용되어 개인의 사생활 침해 및 사회·경제적으로 막대한 피해를 입기도 함. 또한, 유출된 정보는 해킹, 스미싱, 파밍, 명의 도용 등과 같은 또 다른 범죄 행위에 이용되어 심각한 사회 문제를 만들기도 함.

* 해킹과 해커

- 해킹 : 사용자의 허가 없이 시스템에 침투하는 모든 행위
- 해커 : 해킹을 시도하는 사람

I - 3. 정보 보안은 왜 필요할까

1. 정보 보안

정보 보안 : 정보 불법 사용, 해킹, 악성 프로그램, 인터넷 사기, 개인 정보 유출 등 각종 위협으로부터 중요한 정보를 안전하게 보호하는 기술. → 정보의 3요소를 유지하는 기술

정보의 3요소 : 정보를 안전하게 보호하기 위해 제공되어야 하는 것. 정보의 기밀성, 무결성, 가용성이 있음.

기밀성 (공개·노출로부터 보호)	정보의 소유자가 원하지 않으면 정보 공개 X → 허가된 사용자에게만 정보를 제공함
무결성 (변조, 파괴로부터 보호)	정보는 훼손되거나 변형되지 않고 원형 그대로 유지하면서 보존되어야 함. → 허락받지 않은 사용자의 정보 변경 X
가용성 (파괴, 거부, 재난, 지연으로부터 보호)	적법한 사용자, 즉 허락된 사용자가 원할 때에는 언제든지 정보를 제공함.

2. 정보 보안의 필요성 & 3. 정보 보안 실천하기

인터넷을 통한 공격 기술, 크래킹 기술의 발달, 정보 시스템의 취약점 증가, 보안 대책 미비
→ 보안 침해 사고(ex. 스미싱 문자, 파밍*, 고객 정보 유출) 증가
→ 비밀번호 관리, 백신 설치 및 업데이트, 접근 권한 설정, 운영체제 보안 업데이트, 방화벽 사용 및 설정 등으로 사이버 공격으로 인한 피해 줄임

*파밍 : 특정 사이트의 주소를 정확하게 입력해도 가짜 페이지 주소로 연결되어 개인 정보 유출은 물론 물질적 피해도 입힐 수 있는 수법

*DDoS : 분산 서비스 거부. 자료와 정보를 유출하는 것이 목적이 아닌 지속적으로 운영이 필요한 웹사이트에 여러 대의 좀비 PC를 이용하여 한꺼번에 접속해 공격을 가함. → 서버나 네트워크 마비

** 악성 프로그램의 종류 및 특징

바이러스	정상적인 프로그램에 특정 코드를 삽입하여 감염 → 그 프로그램이 실행될 때 바이러스도 함께 실행되어 중요한 정보를 삭제하거나 시스템 자체를 파괴시킴
웜	바이러스와는 달리 다른 프로그램 없이 스스로를 복제하여 통신망을 통해 대량의 복사본을 전송함. 시스템 자체 파괴 X, 통신망 속도 저하
트로이 목마	정상적인 프로그램으로 위장하여 시스템에 설치된 후, 사용자 몰래 중요 정보 유출 or 파일 삭제. 복제 X → 그 프로그램만 지우면 문제 해결
스파이 웨어	사용자 몰래 설치 → 상업적인 목적으로 사용자의 정보 수집, 원치 않는 광고 반복적 노출(애드웨어)
랜섬웨어	컴퓨터 사용자의 파일들을 암호화 하여 그것을 담보로 금전을 요구함. (=멀웨어)

I - 4. 소프트웨어 저작권은 어떻게 보호할 수 있을까

1. 소프트웨어 저작권의 이해

저작물 : 시, 소설, 음악, 미술, 영화, 컴퓨터 프로그램 등에서 사람의 생각이나 감정을 표현한 창작물
저작권 보호를 받는 저작물

어문 저작물	서적, 강연, 연설문 등	건축 저작물	건축물, 설계도 등	도형 저작물	자도, 모형 등
음악 저작물	가사, 악보 등	사진 저작물	독창적·미적 사진 등	2차 저작물	번역, 편곡 등
연극 저작물	연극, 무용, 공연 등	영상 저작물	동영상, 영화 등	편집 저작물	백과사전, 명시선집 등
미술 저작물	회화, 공예, 포스터 등	컴퓨터 프로그래밍 저작물 (소프트웨어)			

저작권 : 저작물에 대한 창작자의 권리. 저작권 보호를 위해 자신이 저작자임을 나타내는 표시를 할 수 있음.

2. 소프트웨어 보호 제도

소프트웨어는 크게 상용 소프트웨어, 공개 소프트웨어, 오픈 소스 소프트웨어, 번들 소프트웨어, 세어웨어로 구성되어 있음.

상용 소프트웨어 : 소프트웨어에 대한 이용 허락의 조건으로 일정한 금액을 지불해야 함

공개 소프트웨어 : 누구나 자유롭게 무료로 사용 할 수 있는 소프트웨어

번들 소프트웨어 : 컴퓨터를 구매할 때 함께 묶여서 판매되는 소프트웨어

세어웨어 : 체험판 또는 평가판으로, 홍보를 위해 일정 기간 사용해 보고 이후에 필요하면 비용 지불

오픈 소스 소프트웨어 : 누구나 이용, 복제, 수정, 배포 등을 자유롭게 할 수 있음. 오픈 소스 소프트웨어를 이용하기 위해서는 라이선스*를 지켜야함.

*라이선스 : 개발자가 규정한 일정 조건 (저작권 표시, 소프트웨어 공개 여부 등)

GPL 라이선스	어떤 목적, 어떤 형태로든 사용 가능, 변경된 프로그램 배포시 동일 라이선스 사용 상용 소프트웨어 사용 불가 (ex. 리눅스, 워드프레스)
MPL 라이선스	소프트웨어를 배포할 경우 저작권, 보증 책임 X, MPL 의해 배포됨 명시 상용 소프트웨어 사용 불가
Apache 라이선스	소프트웨어 배포 시 저작권, 보증 책임 X 내용 표시, 소스코드 공개 의무 X, 'Apache' 상표권 침해 X 상용 소프트웨어로 사용 가능함 (ex. 안드로이드, 휘둠)
BSD 라이선스	소프트웨어 배포시 저작권 표시, 보증 책임 X 내용 표시, 소스코드 공개 의무 X 상용 소프트웨어로 사용 가능함.

I - 5. 사이버 윤리는 어떻게 실천할까

1. 사이버 윤리

사이버 윤리 : 사이버 공간에서 지켜야 할 올바른 가치관과 행동 양식.

사이버 범죄 : 사이버 공간에서의 각종 범죄. ‘익명성’과 ‘비대면성’의 특징을 가지고 있음.

정보 통신망 침해 범죄	- 해킹 - 분산 서비스 거부(DDoS) 공격 등 - 악성 프로그램
불법 콘텐츠 범죄	- 사이버 음란물 - 사이버 도박 - 사이버 명예훼손, 모욕, 사이버 스토킹
정보 통신망 이용 범죄	- 인터넷 사기 (직거래 사기, 쇼핑몰 사기, 게임 사기 등) - 사이버 금융 범죄 (피싱, 파밍, 스미싱, 메모리 해킹 등) - 개인 정보, 위치 정보 침해 - 사이버 저작권 침해 - 스팸 메일

* 피싱 : 타인의 개인 정보를 부정하게 빼내는 사기 수법

* 메모리 해킹 : 피해자의 컴퓨터 메모리에 숨어 있는 악성 코드로 인해 정상적인 은행 사이트 접속하여 보안 카드 번호 앞뒤 2자리만 입력해도 계좌에 있는 돈이 부당하게 인출되는 수법

사이버 범죄는 정보 통신망의 특성상 국내와 국외에서 동시에 발생하기도 하며, 정보 통신망 침해 범죄가 정보 통신망 이용 범죄나 불법 콘텐츠 범죄로 이어지는 등 복합적인 형태로 발생하는 경우가 많음.

2. 사이버 범죄의 예방과 사이버 윤리 실천

사이버 범죄는 범죄가 일어나기 전에 예방하는 것이 중요하므로 사이버 윤리를 지키는 노력을 해야함.

III - 1. 문제란 무엇이고, 어떻게 해결할까

1. 정보과학과 문제

정보과학에서는 다양한 문제 상황을 빠르고 정확한 계산 능력을 가진 컴퓨터를 이용하여 해결하는 것을 목표로 함.

2. 계산 문제

계산 문제 : 컴퓨터를 이용하여 절차적으로 해결할 수 있는 문제들

계산 문제 O	- 15부터 127까지의 자연수 중에서 홀수의 개수 - 미로를 빠져 나오려면? - 하노이의 탑에서 원판을 옮기는 방법은?
계산 문제 X	- 현재 지구의 인구는 70억명을 돌파하였다. 앞으로 10년 후, 인구의 수를 구해보자 - 철수는 현재 지하철을 타고 이동 중이다. 같은 객실에 타고 있는 사람들 중 키가 작은 사람은 몇명인지 구해보자. - 길동이는 마트에서 사과와 배를 샀다. 더 맛있는 과일은 무엇인지 선택해보자.

* 통계는 계산 문제이지만, 통계를 통한 예측은 계산 문제가 아님

인공 지능 - 인간의 지능으로 할 수 있는 학습, 추론, 지각, 자연 언어의 이해 능력 등을 컴퓨터가 할 수 있도록 하는 방법을 연구하는 컴퓨터과학의 한 분야.

→ 인공지능과 정보 통신 기술의 융합으로 미래 사회는 4차 산업 혁명의 시대가 될 것임.

3. 컴퓨터로 문제 해결

컴퓨터를 이용하여 계산 문제를 해결하기 위해서는 문제를 추상화하고 알고리즘을 설계하며, 이를 자동화하는 과정이 필요함.

추상화	주어진 문제를 이해, 분석하여 문제 해결을 위해 불필요한 요소를 제거하거나 작은 문제로 나눔
알고리즘 설계	글이나 도형과 흐름선 등으로 구체적인 문제 해결 방안을 단계별로 나열함 (순서도 작성)
자동화	문제의 해결책(알고리즘)을 컴퓨터 시스템 스스로가 수행할 수 있도록 변환하는 프로그래밍 과정

III - 2. 문제 분석은 어떻게 해야 할까

1. 문제 분석을 통한 현재 상태와 목표 상태 구하기

문제 분석 : 문제 상황을 분석하여 현재 상태(처리되기 전 자료, 혹은 문제)와 목표 상태(적절한 형태로 처리하여 원하는 자료로 만듦)를 파악하고, 수행 작업을 정의하는 과정

문제 분석은 보통 이러한 형태로 이루어짐.

현재 상태 설정하기 → 목표 상태 설정하기 → 수행 작업

2. 핵심 요소 추출

핵심 요소 : 문제를 해결하는 데 있어서 반드시 고려해야 하는 요소

III - 3. 문제 분해와 모델링은 왜 하는 것일까

1. 문제의 분해

문제의 상태를 파악하고 핵심 요소를 추출한 후 다음에 해야 할 일은 주어진 문제를 분해하는 것임.

복잡한 문제 - 한번에 해결 어려움 → 해결 가능한 작은 단위의 문제들로 분해하는 과정 필요

문제를 분해하는 두 가지 방법

1	주어진 문제를 유사한 기능이나 자료를 바탕으로 분해하고, 유사한 작업들을 모아 순차적으로 처리함.
2	주어진 문제를 동일한 형태의 작은 문제로 분해함 → 작은 문제들을 반복적으로 해결하여 주어진 문제 해결

2. 모델링

모델링 : 복잡한 문제를 쉽게 이해하여 해결할 수 있는 형태로 다시 표현하는 것.

텍스트 기반 모델링과 그림이나 도표 형태로 모델링이 있음.

→ 문제의 상태 분석, 핵심 요소 추출 및 분해의 작업 선행 시 더 나은 문제 해결 모델 생성 가능

III - 4. 문제 해결을 위한 알고리즘은 어떻게 설계할까

1. 알고리즘의 이해



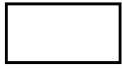

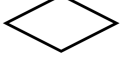
알고리즘 : 문제를 해결하기 위한 구체적인 방법과 절차. 알고리즘은 구체적이면서도 명확하게 표현되어야 함.

2. 알고리즘의 표현

알고리즘은 그것을 읽는 모든 사람이 동일한 작업을 수행할 수 있도록 모호하지 않게 표현해야 함.

알고리즘을 표현하는 방법에는 순서도, 의사 코드, 프로그래밍 언어, 자연어 등이 있음.

* 순서도 : 미리 약속된 도형과 흐름선을 사용하여 알고리즘을 표현하는 방법

	시작 · 끝		흐름
	처리		출력
	비교 · 판단		

3. 알고리즘의 기본 구조

알고리즘은 기본적으로 순차 구조, 반복 구조, 선택 구조를 사용하여 실행 순서와 절차를 나열하여 설계한다.

순차 구조	알고리즘의 실행 순서를 위에서 아래로 순차적으로 표현한 구조를 말함.
반복 구조	필요에 의해 몇 개의 단계를 반복적으로 수행해야 할 때 사용하는 구조
선택 구조	두 개 이상의 값을 비교 → 결과값이 참 또는 거짓인가를 판단 → 다음에 수행할 단계를 결정하는 구조

4. 알고리즘의 설계

모든 문제에 대한 알고리즘을 다 알 수는 없음. → 스스로 알고리즘을 설계할 수 있어야 함.

→ 주어진 문제에 대하여 자신만의 알고리즘을 설계할 줄 아는 것이 많은 알고리즘을 알고 있는 것보다 중요함.

III - 5. 어떤 알고리즘이 더 효율적일까

1. 알고리즘의 효율 분석 ~ 2. 알고리즘의 효율 비교

동일한 문제라도 알고리즘은 여러 가지 있음 → 가장 효율적인 알고리즘을 선택해야 함.

→ 알고리즘의 성능을 수행 시간의 관점(=실행하는 명령의 수)에서 분석하는 것

III - 6. 프로그램은 어떻게 작성할까

1. 프로그램 개발 환경 ~ 2. C 언어 통합 개발 환경

프로그래밍을 하기 위해서는 소스 코드 편집기, 컴파일러, 디버거 등의 도구가 필요함

소스 코드 편집기	프로그램을 작성하는 도구, 프로그램 편집 및 수정 등을 할 수 있음
컴파일러	소스 코드를 컴퓨터가 이해할 수 있는 언어로 번역해 줌
디버거	프로그램을 실행했을 때 오류가 발생하면 수정할 수 있음

→ 통합 개발 환경(IDE) : 소스 코드 편집기, 컴파일러, 디버거 등의 도구들을 사용자가 편하게 사용할 수 있도록 한 곳에 모아 제공하는 소프트웨어

3. C언어 시작하기

C언어 : 컴퓨터에게 명령을 내릴 때 텍스트를 사용하는 텍스트 기반의 프로그래밍 언어 중 하나.

#include <stdio.h> : printf(), scanf() 등 여러 기본적인 함수를 사용하기 위한 헤더 파일(stdio.h)을 불러옴

```
#include <stdio.h>
main(){
    printf("Hello, World");
}
```

실행 결과

```
Hello, World
```

문자열("") : 큰 따옴표로 묶은 문자들

printf() 함수는 다음의 형식으로 씀.

```
printf("문자열");
```

역슬래시를 활용한 기호

\n	줄 바꿈을 나타내는 기호
\t	Tab 키를 나타내는 기호 (일반적으로 간격을 8칸씩 띄운다)
\b	Back Space 키를 나타내는 기호
\\	'\'를 출력하기 위한 기호

```
#include <stdio.h>
main(){
    printf("Hello,");
    printf("World!\n");
    printf("Hello,\b\tWorld!\n");
    printf("1234567812345678\\");
}
```

실행 결과

```
Hello, World!
Hello  World!
1234567812345678\
```

III - 7. 값의 입력과 저장, 연산과 출력은 어떻게 프로그래밍할까

1. 변수와 연산자

상수 : 7, 3.14, 'A' 등과 같이 항상 정해진 값을 가지는 것. 크게 숫자형과 문자형으로 나뉨

숫자형	정수형 상수	소수점이 없는 수를 말함 ex) 3, 13, 0, -21
	실수형 상수	소수점을 포함한 수를 말함 ex) 3.14, 1.021, -2.85, 15.0
문자형	문자 상수	한 문자를 작은따옴표('')로 묶어서 표현한 값 ex) 'a', '#', '6', '!
	문자열 상수	여러 개의 문자를 큰따옴표("")로 묶어서 표현한 값 ex) "Hello World"

변수 : 프로그램을 작성할 때 입력받은 값을 기억하거나 어떤 값을 처리하여 저장하기 위한 기억 장소
변수의 선언은 다음과 같음.

```
자료형 변수명;
```

자료형의 종류와 크기

정수형	int	4바이트
	longlong	8바이트
실수형	float	4바이트
	double	8바이트
문자형	char	1바이트

* 변수명은 영문자, 숫자 0~9, '_'를 포함하여 만들 수 있음. 이때 변수명의 첫 글자는 반드시 영문자
예) 변수명이 a인 정수형 4바이트 변수를 선언하려면

```
int a;
```

연산자 : 산술 연산자, 비교 연산자, 논리 연산자, 대입 연산자 등이 있음.

산술 연산자*	+, -, *, / , %	덧셈, 뺄셈, 곱셈, 나눗셈, 나머지 값을 구할 때 사용함
비교 연산자	>, >=, <, <=, ==, !=	두 개 이상의 값의 크기를 비교할 때 사용함
논리 연산자	&&, , !	AND, OR, NOT과 같이 논리 값을 구할 때 사용함
대입 연산자	=	변수, 배열 등에 특정한 값을 대입할 때 사용함

* C언어의 연산자와 수학 연산자는 다른 부분이 있음! (* = x, / = ÷)

* 나눗셈 연산에서 자료형에 따라 결과값이 다름

(정수형) ÷ (정수형)	5 / 2	2 (정수형 값)
(실수형) ÷ (실수형)	5.0 / 2.0	2.5 (실수형 값)
(정수형) ÷ (실수형) or (실수형) ÷ (정수형)	5 / 2.0 or 5.0 / 2	2.5 (실수형 값)

printf() 함수에서 계산식 or 변수의 값을 출력하고 싶을 때에는 형식 지정 문자를 사용하여야 함

%d	임의의 한 정수값의 형식을 지정함
%c	임의의 한 문자값
%f	임의의 한 실수값(4바이트)의 형식을 지정함
%lf	임의의 한 실수값(8바이트)의 형식을 지정함

예) $5+3$, $32-18$, 21×12 , $12 \div 5$ 의 값을 정수형으로 출력해보자.

```
#include <stdio.h>
main(){
    printf("%d\n", 5+3);
    printf("%d\n", 32-18);
    printf("%d\n", 21*12);
    printf("%d", 12/5);
}
```

실행 결과

```
8
14
252
2
```

2. 입력과 출력

scanf() : 키보드로부터 자료를 입력받기 위해 사용하는 명령문, 변수에 받은 자료를 저장함.

* scanf()와 printf()를 사용하기 위해서는 반드시 stdio.h 헤더 파일을 불러와야 함

```
scanf("%d", &a);
```

scanf()에서 여러 개의 값을 입력할 경우 다음과 같이 입력함 (#은 자신이 원하는 기호로 바꾸기 O)

```
scanf("%d%d", &a, &b);
```

예) 두 정수를 입력받아 덧셈, 뺄셈, 곱셈, 나눗셈을 한 결과값을 각각 출력해보자.

(단, 결과값을 출력할 때 (수)+(수)=(수) 꼴로 출력해보자)

```
#include <stdio.h>
main(){
    int a, b;
    scanf("%d%d", &a, &b);
    printf("%d+%d=%d\n", a, b, a+b);
    printf("%d-%d=%d\n", a, b, a-b);
    printf("%d*d=%d\n", a, b, a*b);
    printf("%d/%d=%d", a, b, a/b);
}
```

실행 결과

```
8 2
10
6
16
4
```

%f와 %lf는 소수점 이하 값을 원하는 자리까지 출력할 수 있음.

예) $13 \div 8$ 의 결과값을 소수점 이하 두자리까지 출력해보자.

```
#include <stdio.h>
main(){
    printf("%.2f", 13.0/8.0);
}
```

실행 결과

```
1.63
```

형식 지정 문자에 숫자를 넣으면 그 자릿수에 맞게 정렬된다.

예) 15×6 의 결과값과 15×12 의 결과값을 천의 자리에 맞춰 출력해보자.

```
#include <stdio.h>
main(){
    printf("%4d\n%4d\n", 15*6, 15*12);
}
```

실행 결과

```
90
180
```

3. 파일을 이용한 입력과 출력

fopen() : 파일을 지정된 모드로 여는 함수. 모드는 두가지가 있으며, r은 읽기, w는 쓰기 모드임.

```
freopen("input.txt", "r", stdin);
freopen("output.txt", "w", stdout);
```

혹은 포인터 변수인 FILE로 파일을 입출력 할 수 있는데, 이를 이용하면 scanf(), printf() 대신 fscanf(), fprintf()를 사용해야 하고, 파일 사용 후에는 fclose()로 닫기!

```
FILE *in = fopen("input.txt", "r");
FILE *out = fopen("output.txt", "w");
fscanf(in, "%d%d", &a, &b);
fprintf(out, "%d\n", a+b);
fclose(in);
fclose(out);
```

예) input.txt에 들어있는 정수를 a에 int형으로 저장하고, a단을 output.txt에 저장해보자.

(단, 포인터 함수를 이용해야 한다.)

```
#include <stdio.h>
main(){
    int a;

    FILE *in = fopen("input.txt","r");
    FILE *out = fopen("output.txt","w");

    fscanf(in,"%d",&a);
    fprintf(out,"%d*1=%d\n",a,a*1);
    fprintf(out,"%d*2=%d\n",a,a*2);
    fprintf(out,"%d*3=%d\n",a,a*3);
    fprintf(out,"%d*4=%d\n",a,a*4);
    fprintf(out,"%d*5=%d\n",a,a*5);
    fprintf(out,"%d*6=%d\n",a,a*6);
    fprintf(out,"%d*7=%d\n",a,a*7);
    fprintf(out,"%d*8=%d\n",a,a*8);
    fprintf(out,"%d*9=%d\n",a,a*9);

    fclose(in);
    fclose(out);
}
```

실행 결과

1.63