

오픈 소스 소프트웨어 기말고사 문제



About..

컴퓨터소프트웨어공학과
김 원 일



기말 실습 시험 준비



- 2인 이상의 팀 구성

- 팀장과 팀원으로 역할을 구성할 2인 이상 4인 이하의 구성원
- 팀장과 팀원에 대한 점수는 동일하기 때문에 역할만을 나눈다고 생각할 것
- 팀 별로 좌석을 잡아 시험 진행에 문제 없도록 할 것

- 시험 PC 정보 설정

- 자격 증명 제거 및 본인 자격 증명과 토큰으로 대체할 것
- Commit에 입력되는 사용자 정보를 자신의 정보로 수정할 것

- git 프로그램 설정

- `git config --global user.name "본인학번"`으로 설정할 것!
- `git config --global user.email "학교메일계정"`으로 설정

- 기타 준비

- github 사이트에 로그인 준비 및 시험에 필요한 프로그램들의 설치



실습 시험 문제 저장소 설정



• 각 반별 문제 저장소

- 1반 : https://github.com/yys-wikim/b1_calc.git
- 2반 : https://github.com/yys-wikim/b2_calc.git
- 팀장은 github로 로그인하여 비어 있는 저장소를 생성
- ReadMe.md, .gitignore, LICENSE 모두 선택하지 않고 저장소만 생성
- 생성할 저장소 이름 : 2025oss
- 팀원을 협력자로 등록하고 시작할 것
- 문제 저장소를 로컬로 복제
- 문제 저장소의 원격 정보를 삭제하고, 2025oss 저장소로 원격 정보 변경
- 저장소 push : `git push --all` 로 모든 브랜치 업로드










실습 시험 팀 분배



• 저장소 상태와 팀원 업무

- main 브랜치를 포함하여 3개의 브랜치가 존재하는 상태
- dev/a, dev/b, dev/c 브랜치를 팀원 수에 맞게 분배
- Ex) 2명이라면 : main과 dev/a를 한명이, 나머지를 한명이 담당
- Ex) 3명이라면 : main과 dev/a를 한명이, 나머지 브랜치를 각각 한명씩 담당
- Ex) 4명이라면 : 모든 브랜치를 각 한명이 담당하여 처리
- 각 팀원이 어떤 브랜치를 작업했는지 README.md에 작성할 것
- 각 팀원은 브랜치에 문제에서 지시된 내용을 그대로 작성하여 Commit

Git Graph					
Branches: Show All		Authors: All		Show Remote Branches	
Graph	Description	Date	Author	Commit	
	1. README.md, images/*.png [*] 화면 구성을 위한 내용과 이미지들을 추가	18 Jun 2025 12:59	wikim	637ec0f6	
	1. calc/dive.h [*] 나누기 함수 원형에 주석 추가	18 Jun 2025 09:59	wikim	296fdaa2	
	1. calc/mod.h, mul.h [*] 나머지, 곱하기 함수 원형 주석 추가	18 Jun 2025 09:58	wikim	e772ab69	
	1. calc/add.h, sub.h [*] 함수 원형 주석 추가	18 Jun 2025 09:57	wikim	ce53e756	
	1. calc/calc.cpp [*] 함수 호출로 결과를 출력하도록 수정	18 Jun 2025 09:54	wikim	47c74979	
	1. add project [+] add project files for calc	18 Jun 2025 08:58	wikim	a57da327	
	Initial commit	18 Jun 2025 08:48	ycs-wikim	061c1751	



실습 시험 문제



• 시험 진행 순서

- 팀장이 업로드한 저장소를 팀원들은 로컬로 복제
- 팀원이 맡은 브랜치에서 코드를 수정하고 브랜치 별로 Commit 후 push
- 각 브랜치의 작업이 모두 완료되어 원격 저장소에 모두 push되었다면
- 팀장이 전체 저장소를 복제하여 main 브랜치에 dev/a, b, c 브랜치를 병합
- 팀원과 팀장이 같이 병합하는 과정에서 발생하는 문제와 해결 방법들을 정리
- 정리하면서 스크린 샷을 필요한 시점에 모두 만들어 둘 것
- 병합이 완료된 코드를 저장소로 push
- 코드 병합이 완료된 프로그램 실행 스크린 샷과 git flow 화면을 캡처
- git flow는 무엇을 이용해도 관계 없으나 터미널에서 직접 출력한 것은 제외!
- GUI 도구에서는 진행 여부를 쉽게 확인할 수 있으나 터미널은 시간이 필요
- main 브랜치에서 README.md를 수정하여 문제 저장소와 같은 형태로 작성
- 작성 완료되면, 원격 저장소로 push 하고 시험을 완료



로컬 복제와 설정



• 팀장만 수행

- `git clone https://github.com/yys-wikim/b1.calc.git`
- **저장소로 이동**
- `git pull --all` 로 모든 정보를 최신으로 설정
- `git branch`로 브랜치 `dev/a`, `dev/b`, `dev/c`가 존재하는지 확인
- 존재하지 않는 경우 각 브랜치를 만들지 말고 이동해서 생성
- `git switch dev/a`
- `git switch dev/b`
- `git switch dev/c`
- 해당 저장소로 즉시 이동되면 문제 없음
- 프로젝트 설정 : Visual Studio 버그로 전체 코드가 포함되지 않는 경우
- 프로젝트가 열린 상태에서 파일을 추가하여 문제를 해결



실습 시험 문제 제출



- e-class로 제출

- 현재 작업을 완료한 디렉터리를 그대로 압축하면 불필요한 파일까지 포함됨
- 제출할 때는 완료된 저장소를 새로운 디렉터를 만들어 복제
 - 불필요한 파일이나 중간 파일들을 모두 제외하기 위함
- 복제된 디렉터를 압축하고, 파일이름을 "팀장_학번.zip"으로 변경
- 압축 파일을 e-class로 팀장만 제출



main 브랜치



- calc.cpp 수정

- 첫번째 입력 값이 -999 가 입력되면 반복을 종료하도록 변경하고 Commit
- Commit 후 원격 저장소로 Commit한 내용을 push

```
int main()
{
    int x = 0;
    int y = 0;
    int i = 0;

    for (; i < 10; i++)
    {
        std::cout << "첫번째 수를 입력하세요 : ";
        std::cin >> x;

        if (-999 == x)
        {
            printf("프로그램을 종료합니다.\n");
            break;
        }

        std::cout << "두번째 수를 입력하세요 : ";
        std::cin >> y;

        printf("입력된 수 x[ %d ] y[ %d ]\n", x, y);
        printf("add[ %d ] sub[ %d ] mul[ %d ] dive[ %d ] mod[ %d ]\n\n",
            add(x, y), sub(x, y), mul(x, y), dive(x, y), mod(x, y));
    }
}
```




- add.cpp, sub.cpp 수정
 - 각 파일을 정상 동작하도록 수정하고 Commit
 - Commit 후 원격 저장소로 Commit한 내용을 push

add.cpp

```
#include <iostream>

int add(int x, int y)
{
    printf("x[ %d ]와 y[ %d ] 값을 더합니다.\n");
    return x + y;
}
```

sub.cpp

```
#include <iostream>

int sub(int x, int y)
{
    printf("x[ %d ]에서 y[ %d ] 값을 뺍니다.\n");
    return x - y;
}
```



- mul.cpp, mod.cpp 수정
 - 각 파일을 정상 동작하도록 수정하고 Commit
 - Commit 후 원격 저장소로 Commit한 내용을 push

```
#include "add.h"
```

```
int mul(int x, int y)
{
    int sum = 0;

    for (int i = 0; i < y; i++)
    {
        sum = add(sum, x);
    }

    return sum;
}
```

mul.cpp

```
#include "sub.h"
```

```
int mod(int x, int y)
{
    while (x > y)
    {
        x = sub(x, y);
    }

    return x;
}
```

mod.cpp



- dive.cpp
 - 파일을 정상 동작하도록 수정하고 Commit
 - Commit 후 원격 저장소로 Commit한 내용을 push

```
#include "sub.h"

int dive(int x, int y)
{
    int cnt = 0;
    while (x > y)
    {
        cnt++;
        x = sub(x, y);
    }

    return cnt;
}
```

All things are difficult, before they are easy.

ReadMe.md 파일

- 기본 README.md 파일 수정
 - 문제가 있는 저장소의 README.md 파일을 참고하여 동일하게 작성
 - 작성된 파일을 Commit 후 원격 저장소로 push

b1_calc

오픈소스 소프트웨어 기말고사 1반 문제

calc

oss 기말 프로젝트

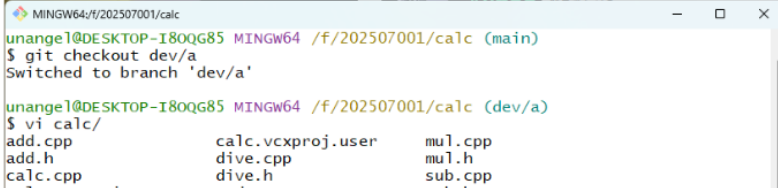
팀원(역할)	업무
홍길동(팀장 - 202407001)	dev/b 브랜치와 main 브랜치 수정
김원일(팀원 - 202407002)	dev/a, dev/c 브랜치와 ReadMe.md 수정

문제해결 방법과 순서

1. main 브랜치와 dev/a 브랜치 병합
2. main 브랜치와 dev/b 브랜치 병합중에 충돌 발생
3. 충돌 발생한 dev/b의 내용을 수정하고 fast-forward 병합 완료
4. dev/c 브랜치를 병합중에 충돌 발생
5. main 브랜치의 내용을 수정하고 rebase 병합 완료
6. 결과 화면 캡처와 실행 화면 캡처
7. readme.md 수정

중간과정 스크린샷

1. 팀원들이 각자 코드를 수정



```
MINGW64~/202507001/calc
unangle1@DESKTOP-I80QG85 MINGW64 /f/202507001/calc (main)
$ git checkout dev/a
Switched to branch 'dev/a'

unangle1@DESKTOP-I80QG85 MINGW64 /f/202507001/calc (dev/a)
$ vi calc/
add.cpp          calc.vcxproj.user      mul.cpp
add.h            dive.cpp                mul.h
calc.cpp         dive.h                  sub.cpp
```

git flow : 결과 화면



The diagram illustrates the Git Flow workflow with branches like main, dev, and feature branches. It shows the sequence of commits, merges, and pull requests, including updates to README.md and other files.

프로그램 실행 결과 화면



```
Microsoft Visual Studio 디버그 x + -
첫 번째 수를 입력하세요 : 8
두 번째 수를 입력하세요 : 2
입력된 수 x[ 8 ] y[ 2 ]
add[ 10 ] sub[ 6 ] mul[ 16 ] dive[ 4 ] mod[ 0 ]

F:\202507001\calc\x64\Debug\calc.exe(프로세스 9780)이(가) 0 코드(0x0)와 함께 종료되었습니다.
이 창을 닫으려면 아무 키나 누르세요...
```