

슈어소프트테크(주)
시험자동화연구소 CS-Static팀
인턴 최가온

SURESOFT

소프트웨어로 안전한 세상을 꿈꾸다

Software for safe world

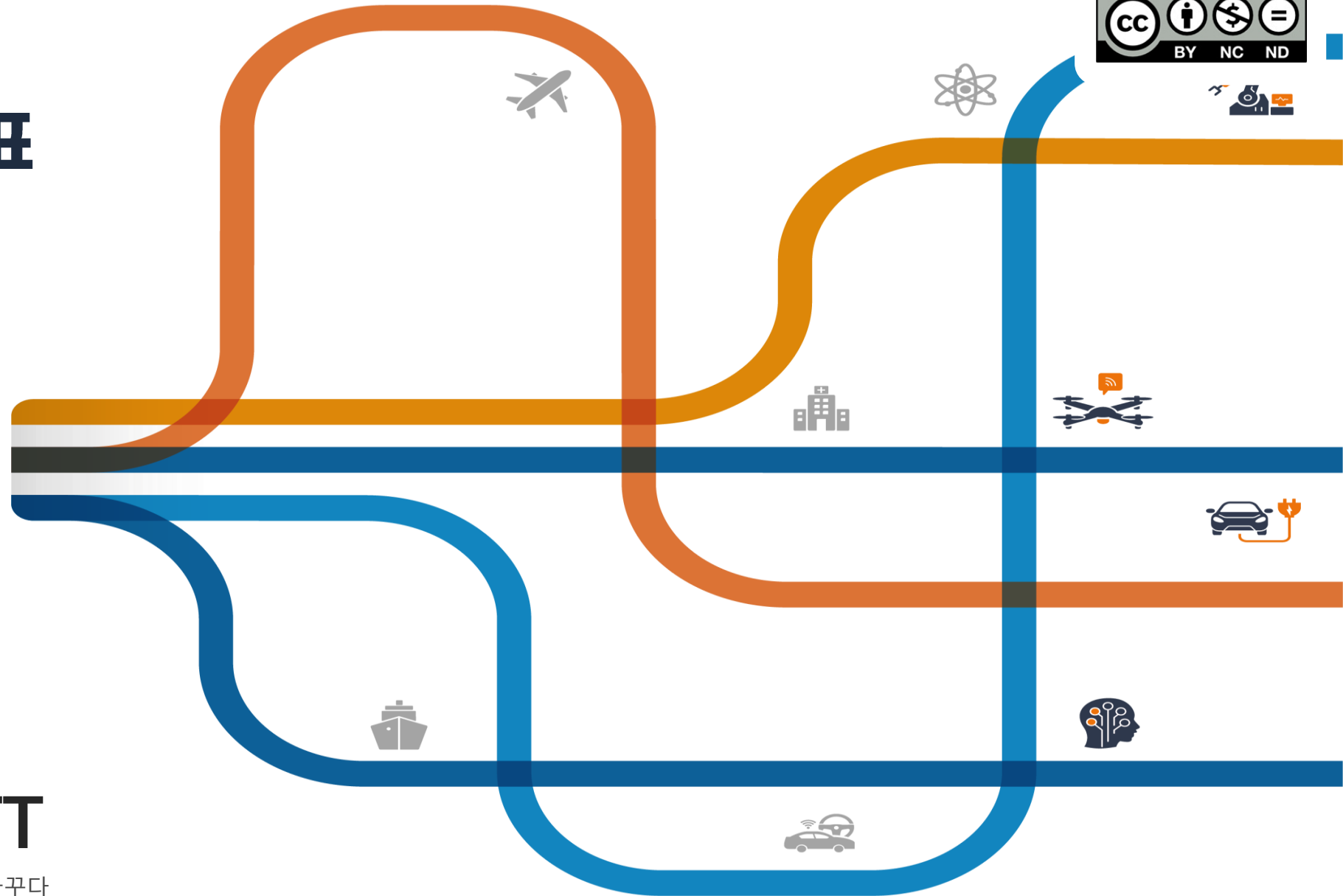


Table of contents

Start

지원 과정

직무 소개

2달 간의 기록

~ing

각 주차별 업무 수행과정

End

업무 성과

느낀점 및 마무리

지원 과정

지원 과정

지원 자격

1. Python 사용 경험자, Web Front End 개발 경험자
2. C/C++ 사용 경험자
3. 기타: Selenium 사용 경험자 혹은 Selenium으로 Web Front-End 테스트 경험자
정적 분석 도구 사용 경험자, 능숙한 영문 번역, C/C++ 표준과 컴파일러에 대한 이해

실습 내용

1. 정적 분석 규칙 예제를 통해 정적 분석 규칙을 숙지 및 데이터를 수집하여 시스템에 추가
2. 자사 도구 (Codescroll STATIC) 사용법 숙지
3. MISRA 규칙 스터디
4. 규칙 Good/Bad 케이스 Codescroll STATIC에 추가
5. 정리 및 세미나와 인수인계

지원 과정

소프트웨어 테스트

“소프트웨어는 언제나 개선의 여지가 남아있으며, 아직 발견하지 못할 버그의 존재 가능성이 있다.”

Static Analysis

실제 실행 없이 소프트웨어를 구성하는 소스 코드를 테스트하는 기법

Dynamic Analysis

실제 또는 가상 프로세서 내에서 프로그램을 실행함으로써 소프트웨어를 테스트하는 기법



지원 과정

소프트웨어 테스트

“소프트웨어는 언제나 개선의 여지가 남아있으며, 아직 발견하지 못할 버그의 존재 가능성이 있다.”

- i) 고려하지 못한 케이스 (요구사항 누락)
- ii) 기대하지 않은 동작 (요구사항 오해)
- iii) 예상치 못한 입력 (예외 처리에 대한 누락)
- iv) 단순한 코딩 실수로 인한 버그



동적 분석

유지 보수의 비용이 큼



정적 분석

시간, 비용이 작음



직무 소개

직무 소개

코딩 규칙 검사

도메인별로 준수해야 하는 코딩 규칙을 자동으로 검사

※ 이미지 출처: 슈어소프트테크(주)



자동차

MISRA C/C++

자동차 내장형 시스템 국제 표준 코딩 규칙



항공

DO-178C / JSF

FAA 항공 내장형 시스템 국제 표준 코딩 규칙



조선/해양

IEC 61508

전기/전자 시스템 소프트웨어 국제 표준 가이드 라인



철도

IEC 62279 / EN 50128

열차제어 시스템 소프트웨어 국제 표준 가이드라인



전기/전자

IEC 61508

전기/전자 시스템 소프트웨어 국제 표준 가이드 라인



의료

IEC 62304

Medical device software — Software life cycle processes



원자력

IEC 60880

국제 원자력 안정성 표준



보안

CWE / CERT Secure Coding Standards

Common Weakness Enumeration



국방

DAPA

방위사업청 신뢰성 지침안 코딩 가이드라인



기타

Naming, Coding style 가이드라인

직무 소개

코딩 규칙 검사



MISRA C/C++

Motor Industry Software Reliability Association

- “Guidelines for the use of the **C language in vehicle based software**”
- 목적: ISO C 언어로 작성된 임베디드 시스템의 코드 안전성, 호환성, 신뢰성
- 자동차 산업으로부터 시작되어 우주/항공, 의료장비, 국방, 철도 등 다양한 산업에서 적용되는 가이드라인 중 하나

직무 소개

1. Rules' Manual 수정사항 개선

띄어쓰기, 오타자 등 문법적인 요소 검사 및 개선

번역체 또는 설명이 난해한 부분에 대해 가독성 개선

MISRA 원문을 참고하여 부가 설명 추가

2. Good/Bad Case 작성

각각의 규칙들을 제대로 설명하고, 직관적인 Good/Bad Case를 설계

설계한 코드들을 Fix Reference에 추가, Good Case와 Bad Case가 최대한 서로 대응되도록 구성

CodeScroll STATIC으로 코드를 분석했을 때 의도한 부분이 정탐의 결과로 나오는지 확인

2달 간의 기록

2달 간의 기록

2020년도 겨울학기 단기 현장실습
(2020년 12월 21~2021년 02월 28)



실습기관명	슈어소프트테크	부서명	CS-Static팀	실습학기	2020년도 겨울학기 단기 현장실습
주간보고서	작성중	종합보고서	작성중	사진첨부	
설문조사	작성중	현장실습기간	2020-12-28~ 2021-02-27	출석일/결석일	29일/0일

㉠ 주간보고서 작성

01 주간보고서	02 종합보고서 작성	03 실습후기	04 설문조사
* 모든 항목은 필수입니다.			
업무요약 *	<div>정적 분석 규칙 예제 데이터 수집 및 시스템에 추가 / UI 테스트 자동화 시스템 구축 (48/50)</div> <div>※ 업무요약은 현장실습 증명서에 출력되므로 현장실습 주요 내용이 포함되게 작성 요망</div>		
업무요약(영문)	<div>CS_STATIC: collecting examples of static analysis rules and its application for systems (87/100)</div> <div>※ 업무요약(영문)은 현장실습 증명서(영어)에 출력되므로 현장실습 주요 내용이 포함되게 작성 요망</div>		
1주차 (278/300)	<div>회사의 기본적인 업무 문화에 대한 이해와 구성원들과 친밀감을 형성하는 시간을 가졌다. 회사에서 직원들과 업무와 관련한 소통을 하기 위해 사용하는 Line, Slack 등을 가입하여 시스템 설치를 완료하고, 사원 계정을 만드는 등 기본적인 회사 업무 시작을 위한 준비 단계를 완료하였다. 두 달 간의 인턴 기간에 대한 업무계약서를 작성하였으며, 코로나 관계로 인해 비대면을 통한 부서 미팅 시간을 가졌다. 또한, CS-Static 부서 과장님과 함께 추후의 업무와 개인적인 상담 등을 진행할 예정이다.</div>		
2주차 (278/300)	<div>CodeScroll STATIC 내에서 MISRA C++ 2008 코드 규칙들을 하나씩 공부해나갔다. 이는 1학년 2학기에 수강한 창의적소프트웨어설계 과목의 연장선으로 생각해볼 수 있다. 각각의 코드 규칙을 학습한 이후, 오타자 / Bad Case+Good Case 등을 검토해나갔다. 또한 매일 아침 간단한 회의를 진행하고 1월 7일에는 OKR 회의에 참석하며 STATIC 부서에서 진행한 프로젝트를 리뷰하는 모습을 관찰해볼 수 있었다. STATIC 부서 팀장님과 신입 사원 면담을 진행하였다.</div>		
3주차 (292/300)	<div>OKR 회의(2차)에 Zoom을 이용하여 참여하였다. 이 회의는 이전의 회의와는 다르게 부서의 모든 인원과 대전 지점의 직원들까지도 모두 참여한 규모의 회의였다. 물론 기술적인 모든 것들을 이해한다기보다는, 회사의 비전이나 회의방식 등을 눈여겨보며 참여하였다. 이를 보며, 발표 능력과 설득력이 회사 내에서 중요하게 작용한다는 생각이 들었다. 이번주까지 MISRA C++ 2008 문서를 어느 정도 작업을 끝내고, MISRA C 2012 document를 보며 검토를 시작하였다. 수정사항들은 사내 공유문서에 저장해두었다.</div>		
4주차 (243/300)	<div>이번주는 코로나19 관련 사내 방침에 따라 재택근무로 현장실습을 진행하였다. MISRA C 2012의 모든 코딩 규칙들에 대한 검토를 완료하였다. 코드 규칙들에 대한 수정사항을 업로드하고 그것들이 어떻게 반영되는지에 대한 프로세스를 안내받았으며, 추후 필자가 작성한 코드규칙 수정사항들에 대해(Good Case/Bad Case, 오타자 교정, 개념 설명 등) 팀원들과 화상회의를 통해 리뷰하</div>		

2달 간의 기록

01 주간보고서	02 종합보고서	03 실습후기	04 설문조사
<div>* 모든 항목은 필수입니다.</div> <div>- 항목별 최소 400자 이상</div>			
적무명 *	CS-STATIC 정적 분석 및 응용		
직무개요 (497/500) *	<p>모든 소프트웨어는 그것이 문제 없이 잘 동작한다고 하더라도, 항상 개선의 여지가 있고 발견하지 못한 버그가 존재할 수 있다. 이러한 버그를 찾아내고 수정하기 위해서는 그 코드들을 분석할 필요가 있다. 소프트웨어 테스트에는 정적 분석(Static Analysis)과 동적 분석(Dynamic Analysis)이 있다. 슈머소프트테크 CS-Static 부서에서는 정적 분석을 하기 위한 자사 개발 도구인 CodeScroll STATIC을 개발하여 많은 회사들을 고객으로 두고 있다. 실습 기간에는 MISRA CPP, C 규칙문서들을 활용하여 자사 내의 규칙 메뉴얼을 수정하고 개선하는 작업을 진행한다. 또한, 각각의 규칙을 설명하는 Good/Bad Case를 여러가지로 만들어보고, 자사 도구와 Git을 이용하여 그 코드들을 테스트하고 정탐/오탐/미탐 등의 여부를 확인하여 실제 메뉴얼에 수정 내용을 반영한다. 학과의 C/C++, 소프트웨어 테스트 과목의 심화 내용에 해당한다고 볼 수 있다.</p> <div>당초 기관에서 공지된 직무내용과 같습니까? <input checked="" type="radio"/> 예 <input type="radio"/> 아니요</div>		
실습내용 (1181/1200) *	<p>위에서 이야기한 정적분석(Static Analysis)은 내부에서 코딩 규칙을 정하고 있다. 이 코딩 규칙은 도메인별로 준수해야 하는 규칙들이 따로 정해져있는데 여러 규칙들 중 자동차와 관련한 규칙인 MISRA C/C++에 대한 메뉴얼을 검토하였다. 먼저 회사에서 기존에 개발한 고객용 메뉴얼을 검토하였다. 그 규칙들에 대한 공식문서를 찾아보고, 주변 개발자들에게 조언을 구하며 MISRA 규칙들을 검토와 동시에 학습해나갔다. 처음에는 공부를 하며 단순 오타차처럼 쉬운 내용들을 중심으로 검토하였다. 이후에는 각각의 규칙들에 대한 설명이 올바른지, 기존의 원문과 비교하였을 때 더 쉬운 설명들에 대해 고민하는 과정을 가졌다. 엑셀 표에 각각의 수정사항들을 일목요연하게 정리한 후 매주 금요일에 따로 메뉴얼 리뷰 회의 시간을 고정적으로 만들어 그 시간에 해당 내용에 대해 팀원들과 논의했다.</p> <p>이후에는 각각의 규칙들을 설명할 수 있는 Good/Bad 케이스들을 만드는 작업을 시작했다. 각 규칙들을 제대로 설명하고, 직관적으로 이해할 수 있을만한 코드를 작성하는데 초점을 두었다. 이를테면 "errno를 사용하면 안된다."라는 규칙을 설명하기 위해 정수 한 개를 입력받아 그것의 양의 제곱근 값을 계산하는 코드에서 음수가 나왔을 경우를 고려한 케이스를 만들었다. 해당 케이스를 자사 개발 도구 주석 하나인 Fix Reference에 연로로 한 후, 가독성을 포함하여 다양한 코드를 고려하여 Good/Bad 케이스 약 30여 개를 만들었다. 해당 코드가 자사개발도구</p>		
실습결과 및 소감 (1200/1200) *	<p>나에게는 개인적인 의미에서는 "첫 직장생활"이라는 점이 크다. 분명 학생으로서의 역할과 직원으로서의 역할에는 차이가 있기에, 최대한 회사 내에서 내가 근무하는 팀에 어떤 형태로든 기여를 하기 위해 의식적으로 노력했다. 2달 동안 근무한 CS-Static 부서가 맡은 업무의 목표를 이해하고 그 속에서 내가 맡은 업무가 무엇인지를 정확히 이해하고, 또 현재 내가 그 업무를 제대로 수행할 수 있는지 알아볼 수 있는 좋은 시스템이 하나 있었다. 그것은 바로 데일리 미팅이었다. 매일 아침 10시에 팀원들과 회의실에 모여 각각 전날에 한 일과 오늘 할 일을 간단하게 브리핑하는 시간을 가졌다. 좁은 의미에서는 업무 보고였지만, 넓은 의미에서는 전체적인 업무의 방향과 그 속에서 개개인의 업무에 대한 원활한 이해에 적합한 시간이었다고 생각한다.</p> <p>결론적으로 지원 시기에 공고문에 명시된 실습 내용들은 실제로 대부분 해볼 수 있었다. 주로 첫번째 목표였던 "정적 분석 규칙 예제를 통해 정적 분석 규칙을 숙지 및 데이터를 수집하여 시스템에 추가" 작업을 하였었다. 학교에서는 C, C++를 구성하는 기본적인 개념을 학습하고, 그 개념들을 이용하여 해결할 수 있는 간단한 형태의 과제를 해결하는 방식으로 학습활동을 해왔다. 이 실습기간 동안에는, C/C++의 기초적인 배경 지식들을 토대로 MISRA 메뉴얼을 참고하여 자사 내의 메뉴얼을 보완하고 개선하는 업무를 진행하였다. 물론 업무라는 특성이 컸지만, 한편으로는 현 직장생활에서(학부생활에서) 학교현장에서 배우 지식에 더불어 실질적인 내용을 배울 수 있는 것으로 보인다고 개인적으로 생각한다.</p>		
후배들에게 하고싶은 말 (1199/1200) * ※ 실습후기에 공개됨 (실습기간은 볼 수 없음)	<p>첫 직장생활에 대해 글감이었습디다. 자작 직작은 6:30에 출근을 합니다. 다 직작에 비해 출근시간이 조금 빠른데도, 자작 고교 재학시절을 재작하고 이공계 글다 글이다든 것은 좀 조금 긴이였습니다. 출근시간 전에 회사에서는 11층에서 아침 식사를 제공합니다. 통학하느라 아침을 못먹으시는 학생들에게는 좋은 소식일 것 같아요. 제가 느꼈던 것은 일찍 출근하는 것이 생각보다 어려웠고 이 기간을 통해 그런 것들을 조금씩 극복해나갔다는 점입니다. 또 퇴근후에는 무언가를 해야겠다는 결심은 많이 해봤지만 실제로는 실천하지 못한 부분들도 많았습니다. 직장 생활을 하면서는 자기계발을 틈틈이 해야되는데 체력이 무엇보다 중요하다는 것을 새삼 느끼게 되기도 했습니다.</p> <p>업무에 대해 이전에 자세한 소개를 하였지만, 저희 학부의 창소프 시간에 배우는 C/C++의 기초적인 지식만 있으면 가능합니다. 어떻게 코드를 짜야 보안, 기타 성능 등에서 안전하지에 대한 물을 배우고 싶으신 분들에게 추천합니다. 주로 회사부서에 도움이 되지만 동시에 학부생에게도 배울 거리가 있는 업무를 주는 편이라 판단됩니다. 회사 분위기는 전체적으로 수평적입니다. 처음 오시면 소장님과 면담을 할 기회도 주어집니다. 특이한 점은 회사가 정적분석 툴을 개발하기에 주로 제조업 회사가 고객출인데, 2달간 느꼈지만 근무하시는 분들이 나름의 자부심을 갖고 계신것이 보였습니다. 실습기간이 끝나가는 지금 드릴 선물을 고민하고 있는 저의 모습을 보면 개인적으로도 성장하고 만족스러운 생활이었던 것 같습니다.</p>		

각 주차별 업무 수행과정

각 주차별 업무 수행과정 (1주차~2주차)

1. 업무 환경 및 안내사항 숙지

- 노트북, 사내 메신저 (Line), 팀내 메신저 (Slack) 등 업무 환경 조성
- 슈어소프트테크 신입사원 가이드, CS-Static 부서 Notion 신입사원 가이드
- 팀원 얼굴, 이름 익히기
- 1주일 단위로 팀 내에서 하는 활동 파악하기 (Daily meeting, Demo day ...)

2. MISRA CPP 2008 문서 스터디

3. MISRA C 2012 문서 스터디

변수와 배열, 표현식, 함수의 정의, 구조체(Structure), 클래스(class), 템플릿(template) 등

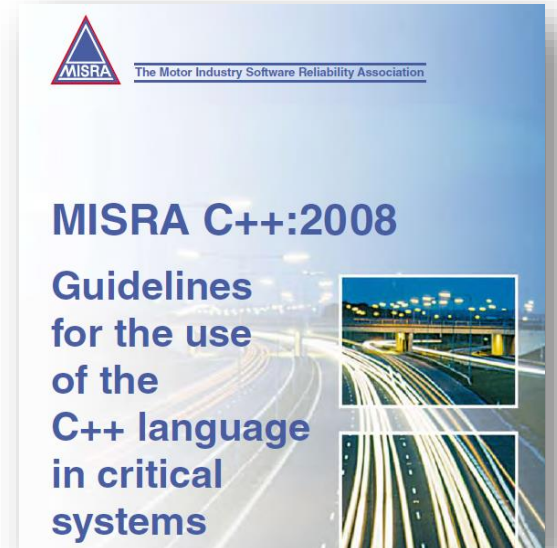
4. CS-Static부서 팀장님, 소장님과의 면담 진행

본격적인 업무를 시작하기에 앞서, STATIC 직무 및 업무내용에 대한 면담

각 주차별 업무 수행과정 (3주차~5주차)

1. MISRA CPP 2008, MISRA C 2012 매뉴얼 개선작업

- 띄어쓰기, 오타자 등의 문법적인 오류 사항 수정
- 번역체로 쓰여진 기존의 이해하기 어려운 설명
- 문의 들어온 규칙 중 정답으로 판정된 케이스들에 대한 설명 보완
- CS 용어의 자연스러운 한글화 작업(Overload Resolution, placement new/free ...)
- 사소한 용어나 표기 전체적으로 통일하여 수정(Compliant 표기 통일 등)



각 주차별 업무 수행과정 (3주차~5주차)

1. MISRA CPP 2008, MISRA C 2012 매뉴얼 개선작업

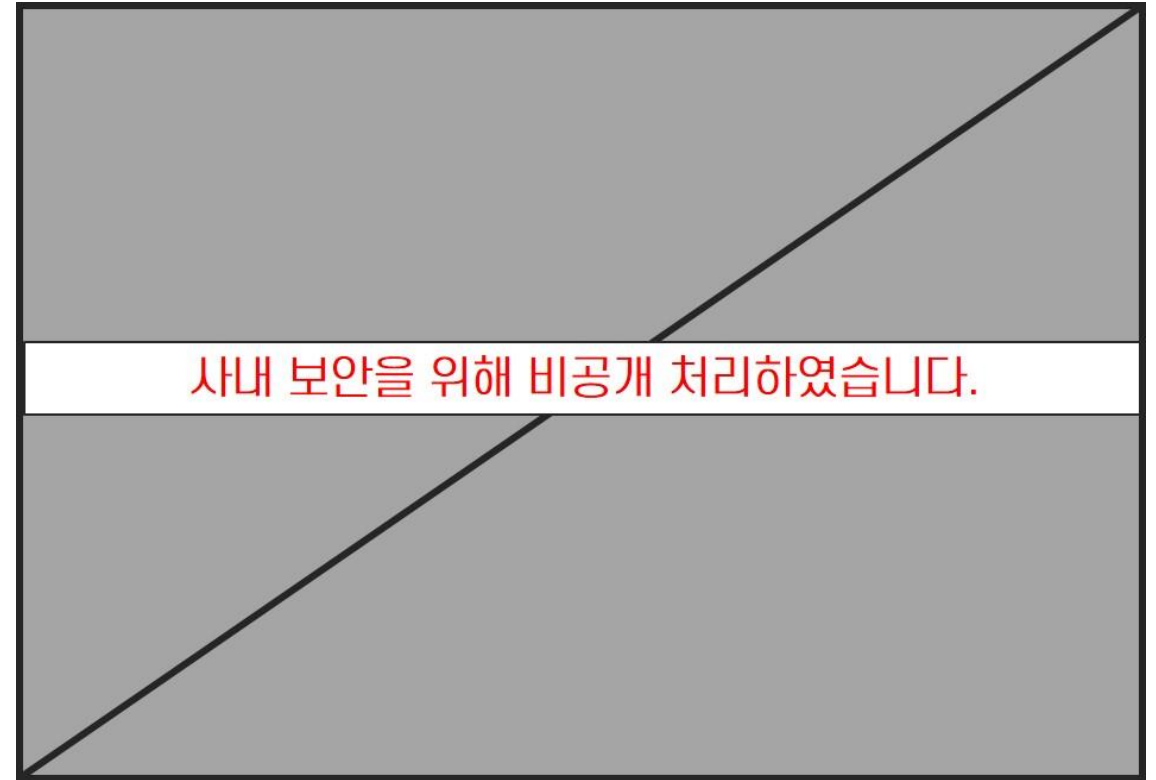
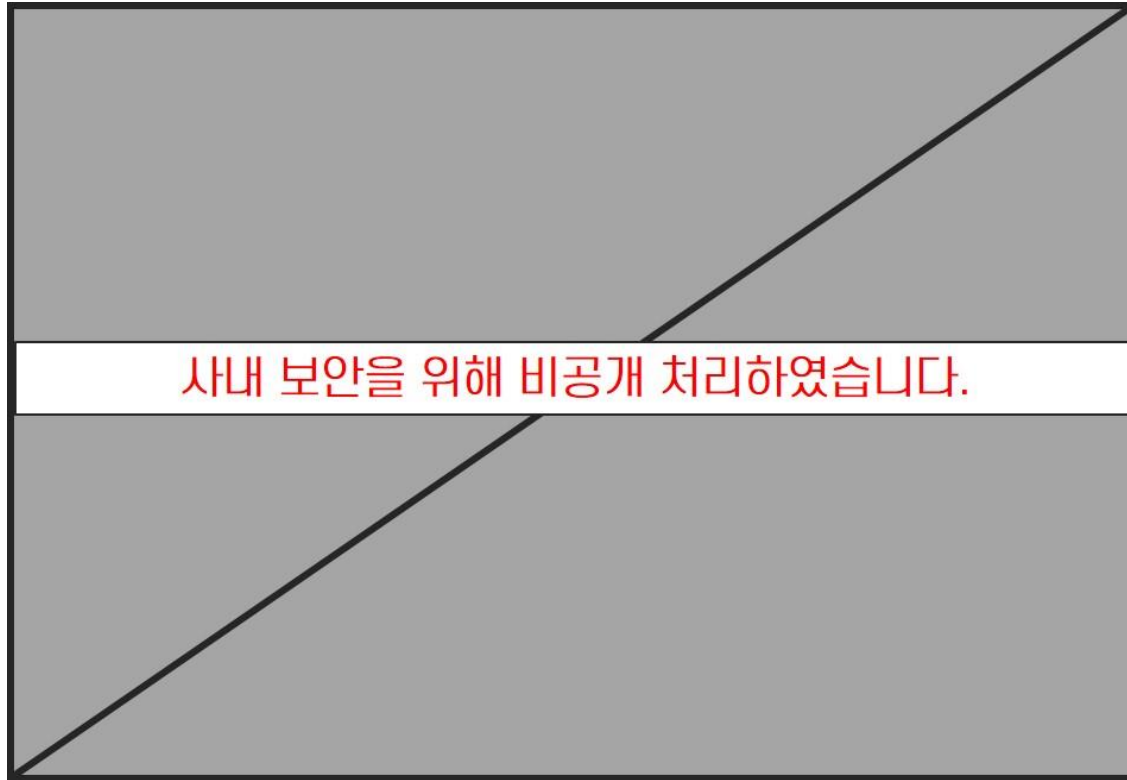
1-1> 띄어쓰기, 오타자 등의 문법적인 오류 사항 수정

[A]		[B]		[C]		[D]		[E]		[F]		[G]		[H]		[I]		[J]		[K]		[L]		[M]		[N]		[O]		[P]		[Q]		[R]		[S]		[T]		[U]		[V]		[W]		[X]		[Y]		[Z]		[AA]		[AB]		[AC]		[AD]		[AE]		[AF]		[AG]		[AH]		[AI]		[AJ]		[AK]		[AL]		[AM]		[AN]		[AO]		[AP]		[AQ]		[AR]		[AS]		[AT]		[AU]		[AV]		[AW]		[AX]		[AY]		[AZ]		[BA]		[BB]		[BC]		[BD]		[BE]		[BF]		[BG]		[BH]		[BI]		[BJ]		[BK]		[BL]		[BM]		[BN]		[BO]		[BP]		[BQ]		[BR]		[BS]		[BT]		[BU]		[BV]		[BW]		[BX]		[BY]		[BZ]		[CA]		[CB]		[CC]		[CD]		[CE]		[CF]		[CG]		[CH]		[CI]		[CJ]		[CK]		[CL]		[CM]		[CN]		[CO]		[CP]		[CQ]		[CR]		[CS]		[CT]		[CU]		[CV]		[CW]		[CX]		[CY]		[CZ]		[DA]		[DB]		[DC]		[DD]		[DE]		[DF]		[DG]		[DH]		[DI]		[DJ]		[DK]		[DL]		[DM]		[DN]		[DO]		[DP]		[DQ]		[DR]		[DS]		[DT]		[DU]		[DV]		[DW]		[DX]		[DY]		[DZ]		[EA]		[EB]		[EC]		[ED]		[EE]		[EF]		[EG]		[EH]		[EI]		[EJ]		[EK]		[EL]		[EM]		[EN]		[EO]		[EP]		[EQ]		[ER]		[ES]		[ET]		[EU]		[EV]		[EW]		[EX]		[EY]		[EZ]		[FA]		[FB]		[FC]		[FD]		[FE]		[FF]		[FG]		[FH]		[FI]		[FJ]		[FK]		[FL]		[FM]		[FN]		[FO]		[FP]		[FQ]		[FR]		[FS]		[FT]		[FU]		[FV]		[FW]		[FX]		[FY]		[FZ]		[GA]		[GB]		[GC]		[GD]		[GE]		[GF]		[GG]		[GH]		[GI]		[GJ]		[GK]		[GL]		[GM]		[GN]		[GO]		[GP]		[GQ]		[GR]		[GS]		[GT]		[GU]		[GV]		[GW]		[GX]		[GY]		[GZ]		[HA]		[HB]		[HC]		[HD]		[HE]		[HF]		[HG]		[HH]		[HI]		[HJ]		[HK]		[HL]		[HM]		[HN]		[HO]		[HP]		[HQ]		[HR]		[HS]		[HT]		[HU]		[HV]		[HW]		[HX]		[HY]		[HZ]		[IA]		[IB]		[IC]		[ID]		[IE]		[IF]		[IG]		[IH]		[II]		[IJ]		[IK]		[IL]		[IM]		[IN]		[IO]		[IP]		[IQ]		[IR]		[IS]		[IT]		[IU]		[IV]		[IW]		[IX]		[IY]		[IZ]		[JA]		[JB]		[JC]		[JD]		[JE]		[JF]		[JG]		[JH]		[JI]		[JJ]		[JK]		[JL]		[JM]		[JN]		[JO]		[JP]		[JQ]		[JR]		[JS]		[JT]		[JU]		[JV]		[JW]		[JX]		[JY]		[JZ]		[KA]		[KB]		[KC]		[KD]		[KE]		[KF]		[KG]		[KH]		[KI]		[KJ]		[KK]		[KL]		[KM]		[KN]		[KO]		[KP]		[KQ]		[KR]		[KS]		[KT]		[KU]		[KV]		[KW]		[KX]		[KY]		[KZ]		[LA]		[LB]		[LC]		[LD]		[LE]		[LF]		[LG]		[LH]		[LI]		[LJ]		[LK]		[LL]		[LM]		[LN]		[LO]		[LP]		[LQ]		[LR]		[LS]		[LT]		[LU]		[LV]		[LW]		[LX]		[LY]		[LZ]		[MA]		[MB]		[MC]		[MD]		[ME]		[MF]		[MG]		[MH]		[MI]		[MJ]		[MK]		[ML]		[MN]		[MO]		[MP]		[MQ]		[MR]		[MS]		[MT]		[MU]		[MV]		[MW]		[MX]		[MY]		[MZ]		[NA]		[NB]		[NC]		[ND]		[NE]		[NF]		[NG]		[NH]		[NI]		[NJ]		[NK]		[NL]		[NM]		[NN]		[NO]		[NP]		[NQ]		[NR]		[NS]		[NT]		[NU]		[NV]		[NW]		[NX]		[NY]		[NZ]		[OA]		[OB]		[OC]		[OD]		[OE]		[OF]		[OG]		[OH]		[OI]		[OJ]		[OK]		[OL]		[OM]		[ON]		[OO]		[OP]		[OQ]		[OR]		[OS]		[OT]		[OU]		[OV]		[OW]		[OX]		[OY]		[OZ]		[PA]		[PB]		[PC]		[PD]		[PE]		[PF]		[PG]		[PH]		[PI]		[PJ]		[PK]		[PL]		[PM]		[PN]		[PO]		[PP]		[PQ]		[PR]		[PS]		[PT]		[PU]		[PV]		[PW]		[PX]		[PY]		[PZ]		[QA]		[QB]		[QC]		[QD]		[QE]		[QF]		[QG]		[QH]		[QI]		[QJ]		[QK]		[QL]		[QM]		[QN]		[QO]		[QP]		[QQ]		[QR]		[QS]		[QT]		[QU]		[QV]		[QW]		[QX]		[QY]		[QZ]		[RA]		[RB]		[RC]		[RD]		[RE]		[RF]		[RG]		[RH]		[RI]		[RJ]		[RK]		[RL]		[RM]		[RN]		[RO]		[RP]		[RQ]		[RR]		[RS]		[RT]		[RU]		[RV]		[RW]		[RX]		[RY]
-----	--	-----	--	-----	--	-----	--	-----	--	-----	--	-----	--	-----	--	-----	--	-----	--	-----	--	-----	--	-----	--	-----	--	-----	--	-----	--	-----	--	-----	--	-----	--	-----	--	-----	--	-----	--	-----	--	-----	--	-----	--	-----	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------	--	------

각 주차별 업무 수행과정 (3주차~5주차)

1. MISRA CPP 2008, MISRA C 2012 매뉴얼 개선작업

1-2> 번역체로 쓰여진 기존의 이해하기 어려운 설명



각 주차별 업무 수행과정 (3주차~5주차)

1. MISRA CPP 2008, MISRA C 2012 매뉴얼 개선작업

1-3> 문의 들어온 규칙 중 정답으로 판정된 케이스들에 대한 설명 보완

(Example) MISRA_C_2012_08_02

추상 클래스에서 복사 대입 연산자는 protected 또는 private으로 선언되어야 한다.

< BEFORE >

추상 클래스는 상속에서 인터페이스 부분을 표현한다. 이런 상속 관계에서 최상위 복사 생성자를 호출하게 된다면 그 하위 클래스들에서 구현해 놓은 구현체들은 무시한 채 기본 클래스의 객체만 복사되게 된다.



< AFTER >

추상 클래스는 상속에서 인터페이스 부분을 표현한다. 이런 상속 관계에서 최상위 복사 생성자를 호출하게 된다면 그 하위 클래스들에서 구현해 놓은 구현체들은 무시한 채 기본 클래스의 객체만 복사되게 된다.

※ 컴파일러에 의해 자동적으로 생성되는 추상 클래스의 디폴트 복사 대입 연산자는 public으로 선언된다. 따라서, 명시적으로 접근 권한을 제한하기 위해 private이나 protected로 변경해야 한다.

각 주차별 업무 수행과정 (3주차~5주차)

1. MISRA CPP 2008, MISRA C 2012 매뉴얼 개선작업

1-4> CS 용어의 자연스러운 한글화 작업

(Example) MISRA_C_14_06_02

Overload Resolution이 식별하는 함수 검사

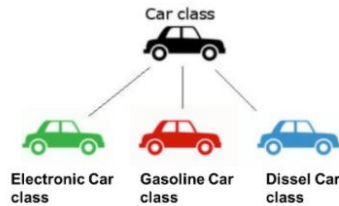
Overload resolution		오버로드 결정 오버로드된 함수를 찾는 과정
Placement new/delete		위치 지정 new/delete
Built-in new/free		내장(內藏) new/delete

각 주차별 업무 수행과정 (3주차~5주차)

1. MISRA CPP 2008, MISRA C 2012 매뉴얼 개선작업

1-5> 사소한 용어나 표기 전체적으로 통일하여 수정

(Example)



기본 클래스, base class
부모 클래스, parent class

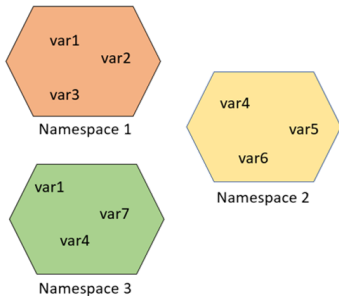


상위 클래스

파생 클래스, derived class
자식 클래스, child class



하위 클래스



네임스페이스
이름공간
name space
...



namespace



/* non-compliant */
/* Not Compliant */



/* Non-compliant */

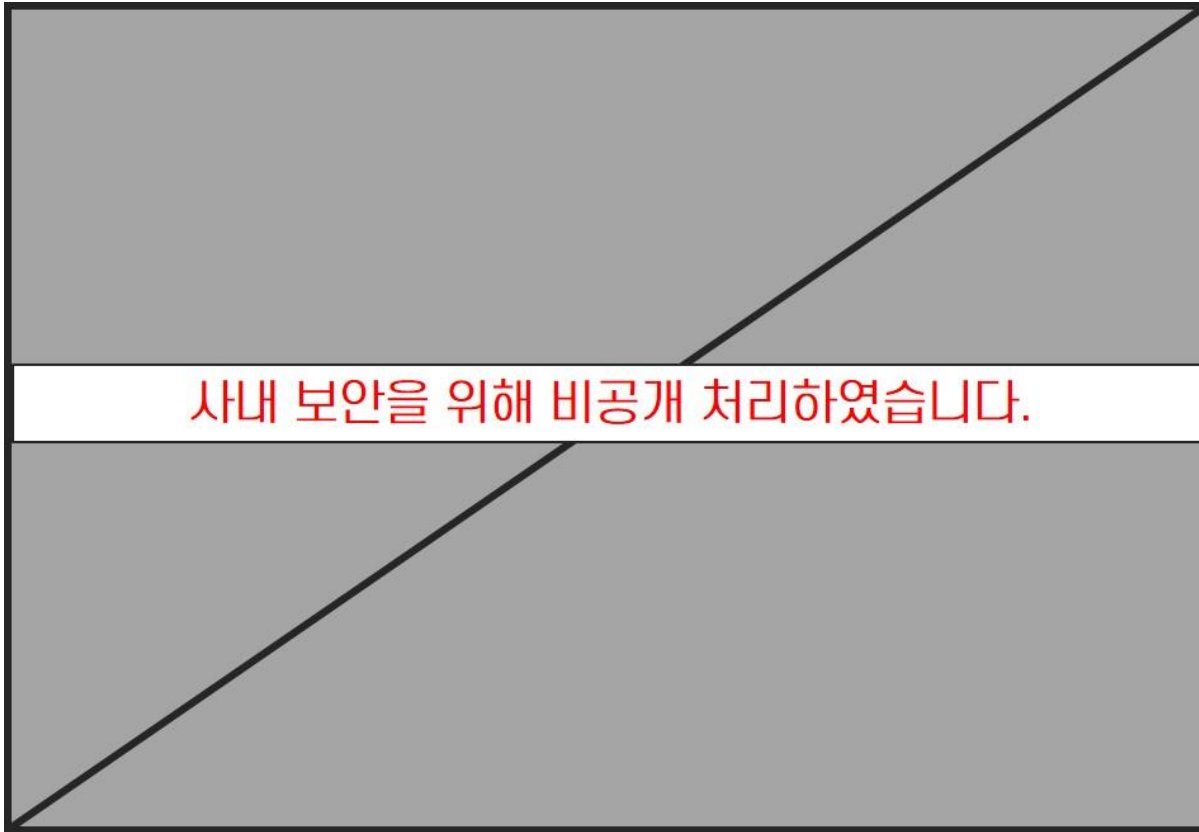
/* compliant */



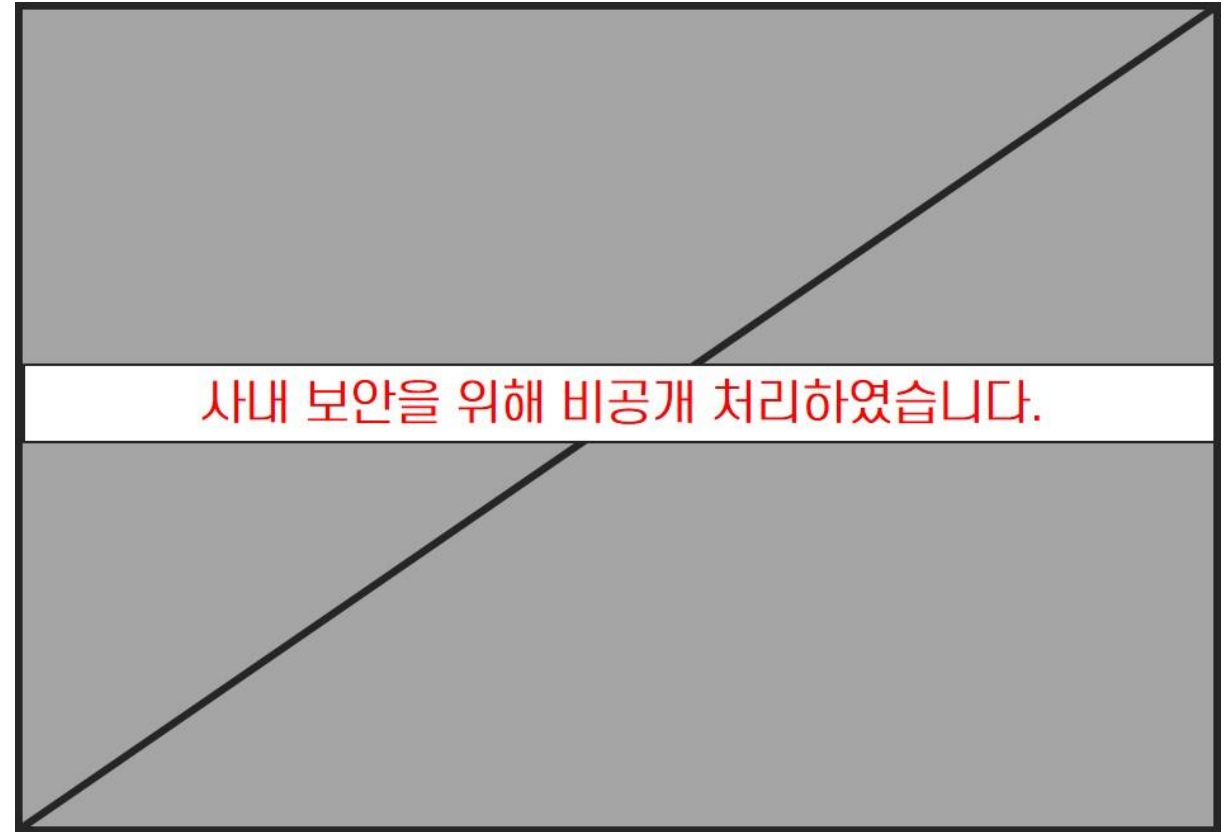
/* Compliant */

각 주차별 업무 수행과정 (3주차~5주차)

2. Git 사용법, Rule Database Generator(RDG) 사용법 숙지



< Git을 이용한 팀 단위 작업 >



< RDG 내에서 개선사항 반영 >

각 주차별 업무 수행과정 (6주차)

1. 문의 들어온 규칙 재검토

MEMO CPP 2000 송환 수									
작성: 송환, 송환, 송환, 송환, 송환, 송환, 송환, 송환, 송환, 송환									
1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100
101	102	103	104	105	106	107	108	109	110
111	112	113	114	115	116	117	118	119	120
121	122	123	124	125	126	127	128	129	130
131	132	133	134	135	136	137	138	139	140
141	142	143	144	145	146	147	148	149	150
151	152	153	154	155	156	157	158	159	160
161	162	163	164	165	166	167	168	169	170
171	172	173	174	175	176	177	178	179	180
181	182	183	184	185	186	187	188	189	190
191	192	193	194	195	196	197	198	199	200
201	202	203	204	205	206	207	208	209	210
211	212	213	214	215	216	217	218	219	220
221	222	223	224	225	226	227	228	229	230
231	232	233	234	235	236	237	238	239	240
241	242	243	244	245	246	247	248	249	250
251	252	253	254	255	256	257	258	259	260
261	262	263	264	265	266	267	268	269	270
271	272	273	274	275	276	277	278	279	280
281	282	283	284	285	286	287	288	289	290
291	292	293	294	295	296	297	298	299	300
301	302	303	304	305	306	307	308	309	310
311	312	313	314	315	316	317	318	319	320
321	322	323	324	325	326	327	328	329	330
331	332	333	334	335	336	337	338	339	340
341	342	343	344	345	346	347	348	349	350
351	352	353	354	355	356	357	358	359	360
361	362	363	364	365	366	367	368	369	370
371	372	373	374	375	376	377	378	379	380
381	382	383	384	385	386	387	388	389	390
391	392	393	394	395	396	397	398	399	400
401	402	403	404	405	406	407	408	409	410
411	412	413	414	415	416	417	418	419	420
421	422	423	424	425	426	427	428	429	430
431	432	433	434	435	436	437	438	439	440
441	442	443	444	445	446	447	448	449	450
451	452	453	454	455	456	457	458	459	460
461	462	463	464	465	466	467	468	469	470
471	472	473	474	475	476	477	478	479	480
481	482	483	484	485	486	487	488	489	490
491	492	493	494	495	496	497	498	499	500
501	502	503	504	505	506	507	508	509	510
511	512	513	514	515	516	517	518	519	520
521	522	523	524	525	526	527	528	529	530
531	532	533	534	535	536	537	538	539	540
541	542	543	544	545	546	547	548	549	550
551	552	553	554	555	556	557	558	559	560
561	562	563	564	565	566	567	568	569	570
571	572	573	574	575	576	577	578	579	580
581	582	583	584	585	586	587	588	589	590
591	592	593	594	595	596	597	598	599	600
601	602	603	604	605	606	607	608	609	610
611	612	613	614	615	616	617	618	619	620
621	622	623	624	625	626	627	628	629	630
631	632	633	634	635	636	637	638	639	640
641	642	643	644	645	646	647	648	649	650
651	652	653	654	655	656	657	658	659	660
661	662	663	664	665	666	667	668	669	670
671	672	673	674	675	676	677	678	679	680
681	682	683	684	685	686	687	688	689	690
691	692	693	694	695	696	697	698	699	700
701	702	703	704	705	706	707	708	709	710
711	712	713	714	715	716	717	718	719	720
721	722	723	724	725	726	727	728	729	730
731	732	733	734	735	736	737	738	739	740
741	742	743	744	745	746	747	748	749	750
751	752	753	754	755	756	757	758	759	760
761	762	763	764	765	766	767	768	769	770
771	772	773	774	775	776	777	778	779	780
781	782	783	784	785	786	787	788	789	790
791	792	793	794	795	796	797	798	799	800
801	802	803	804	805	806	807	808	809	810
811	812	813	814	815	816	817	818	819	820
821	822	823	824	825	826	827	828	829	830
831	832	833	834	835	836	837	838	839	840
841	842	843	844	845	846	847	848	849	850
851	852	853	854	855	856	857	858	859	860
861	862	863	864	865	866	867	868	869	870
871	872	873	874	875	876	877	878	879	880
881	882	883	884	885	886	887	888	889	890
891	892	893	894	895	896	897	898	899	900
901	902	903	904	905	906	907	908	909	910
911	912	913	914	915	916	917	918	919	920
921	922	923	924	925	926	927	928	929	930
931	932	933	934	935	936	937	938	939	940
941	942	943	944	945	946	947	948	949	950
951	952	953	954	955	956	957	958	959	960
961	962	963	964	965	966	967	968	969	970
971	972	973	974	975	976	977	978	979	980
981	982	983	984	985	986	987	988	989	990
991	992	993	994	995	996	997	998	999	1000

사내 보안을 위해 비공개 처리하였습니다.

각 주차별 업무 수행과정 (6주차)

2. Modern C++ 세미나 참석



- `constexpr` / `constexpr` : 컴파일 타임에 계산 가능하면 컴파일 타임에 계산
- `coroutine` : 비동기코드에서의 context switching 오버헤드 문제 경감
- `likely` / `unlikely` : 컴파일러에 분기 최적화의 여지를 제공

(likely – True일 확률 HIGH, unlikely – False일 확률 HIGH)

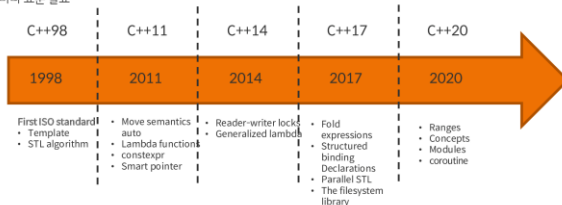
- `concept` : named set of requirements, 매개변수 등에 `require` 키워드로 제약 조건을 설정

모던 C++

`constexpr`과 `constexpr` 3/5

`coroutines` 3/22

- C++ 98로 처음 표준화
- C++ 11 표준 이후 꾸준히 업데이트
- 3년 마다 표준 발표

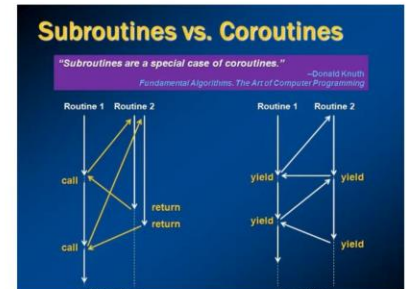


`constexpr`은 무조건 컴파일시간에 계산

```
constexpr int sqr(int n) {  
    return n*n;  
}  
  
constexpr int r = sqr(100); // OK  
  
int x = 100;  
int r2 = sqr(x); // Error: Call does not produce a constant
```

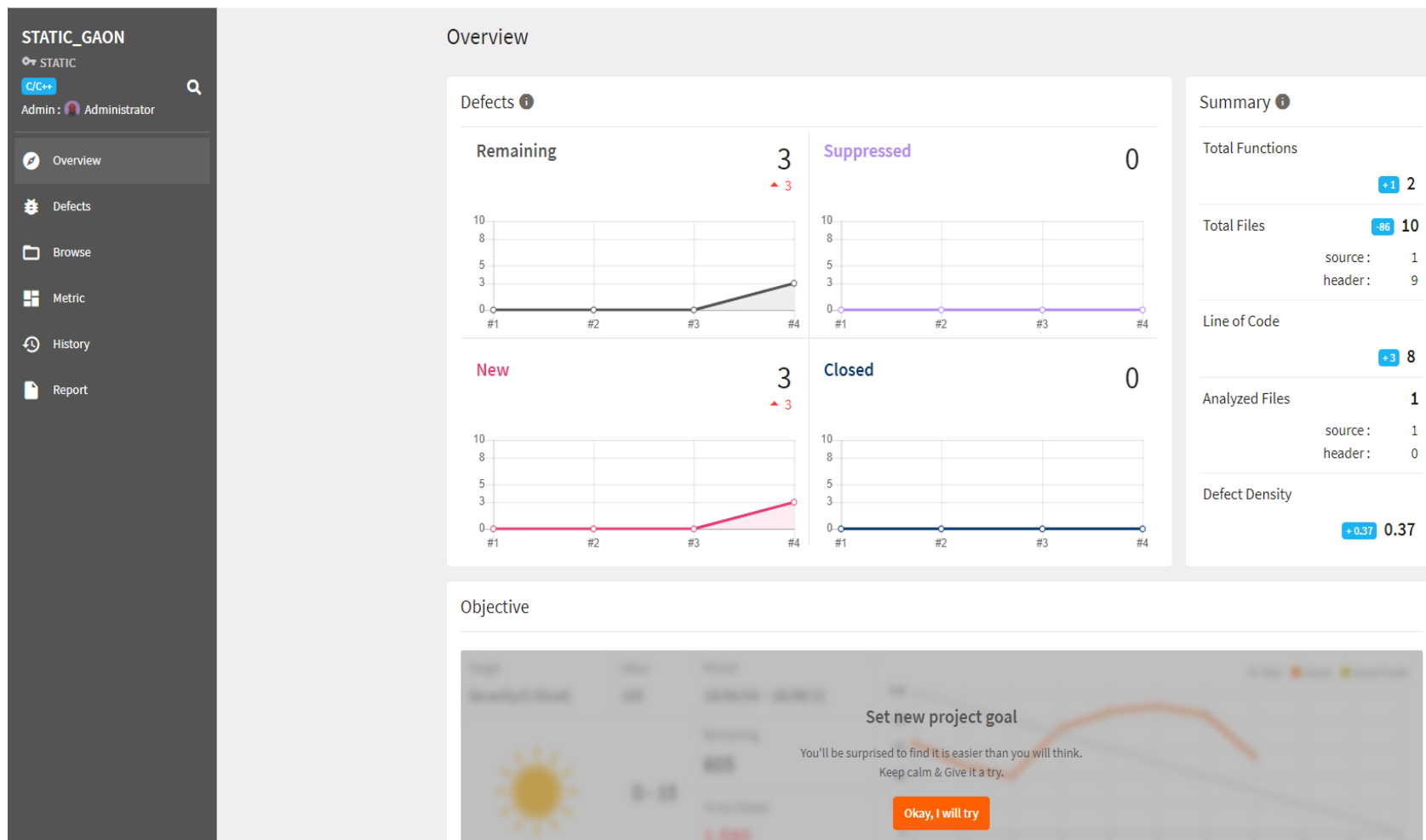
왼쪽은 일반 함수이며 오른쪽이 코루틴

일반 함수와 별개로 중간/재개가 가능



각 주차별 업무 수행과정 (7주차)

1. 정적분석도구 Codescroll STATIC 사용법 숙지



각 주차별 업무 수행과정 (7주차)

1. 정적분석도구 Codescroll STATIC 사용법 숙지

The screenshot displays the Codescroll STATIC static analysis tool interface. On the left is a sidebar with navigation options: Overview, Defects, Browse (selected), Metric, History, and Report. The main area is titled 'Browse' and shows the file path 'C:/USERS/CHOIG/SOURCE/REP... > STATIC_TEST.CPP'. Below this, a list of defects is shown, each with an icon, a description, and a line number. The defects include: 'function real가 전역 namespace에 있음' (Line: 4), 'double basic type 이 사용되었음' (Line: 4), '파라미터 num 가 함수 안에서 사용되지 않았음' (Line: 4), 'void 함수가 external side effect가 없음' (Line: 4), '함수 real의 정의 이전에 선언이 존재하지 않음' (Line: 4), 'int basic type 이 사용되었음' (Line: 8), '함수 main 는 2 개의 exit point를 가짐' (Line: 8), 'int basic type 이 사용되었음' (Line: 10), and '변수 a의 값이 변경되지 않음에도 const로 선언되어 있지 않음' (Line: 17). On the right, the code editor shows the source code for 'STATIC_TEST.CPP'. The code includes headers for <iostream> and <cmath>, defines a function 'real(double num)', and a 'main()' function. A red highlight is placed on line 10, which contains the declaration 'int num;'. A tooltip for this line indicates a 'MISRA_CPP_03_09_02' rule violation: 'int basic type 이 사용되었음'.

STATIC_GAON
STATIC
C/C++
Admin: Administrator

Overview
Defects
Browse
Metric
History
Report

Browse
STATIC > C:/USERS/CHOIG/SOURCE/REP... > STATIC_TEST.CPP

Defect [ALT] + ↑, [ALT] + ↓

- function real가 전역 namespace에 있음 Line: 4
- double basic type 이 사용되었음 Line: 4
- 파라미터 num 가 함수 안에서 사용되지 않았음 Line: 4
- void 함수가 external side effect가 없음 Line: 4
- 함수 real의 정의 이전에 선언이 존재하지 않음 Line: 4
- int basic type 이 사용되었음 Line: 8
- 함수 main 는 2 개의 exit point를 가짐 Line: 8
- int basic type 이 사용되었음 Line: 10
- 변수 a의 값이 변경되지 않음에도 const로 선언되어 있지 않음 Line: 17

C:/USERS/CHOIG/SOURCE/REPOS/STATIC_TEST/STATIC_TEST/STATIC_TEST.CPP

Critical	Major	Minor	Trivial	Other	Line of Code
-	6	4	-	-	16

```
1 #include <iostream>
2 #include <cmath>
3
4 void real(double num) {
5     // ...
6 }
7
8 int main()
9 {
10     int num;
11
12     std::cin >> num;
13     if (num < 0)
14     {
15         return -1;
16     }
17     double a = sqrt(num);
18
19     real(a);
20     return 0;
21 }
22
```

[MISRA_CPP_03_09_02]

STATIC_TEST.CPP Line: 10 ⚡ main

int basic type 이 사용되었음

각 주차별 업무 수행과정 (7주차)

2. Good/Bad Case 설계 및 수정

(Example 1) MISRA_CPP_19_03_01

에러 표시자(indicator) errno는 사용하면 안된다.

sample CHANGED	
<pre>@@ -1,18 +1,19 @@ 1 #include <iostream> 2 #include <cmath> 3 - #include <cerrno> 4 void real(double num) { 5 // ... 6 } 7 int main() 8 { 9 int num; 10 std::cin >> num; 11 - double a = sqrt(num); 12 - if (errno == EDOM) 13 { 14 return -1; 15 } 16 real(a); 17 return 0; 18 }</pre>	<pre>1 #include <iostream> 2 #include <cmath> 3 void real(double num) { 4 // ... 5 } 6 int main() 7 { 8 int num; 9 std::cin >> num; 10 + if (num < 0) 11 { 12 return -1; 13 } 14 + 15 + double a = sqrt(num); 16 + 17 real(a); 18 return 0; 19 }</pre>

각 주차별 업무 수행과정 (7주차)

2. Good/Bad Case 설계 및 수정

(Example 2) MISRA_C_2012_13_04

할당 연산자의 결과를 사용 금지

sample CHANGED	
@@ -1,15 +1,17 @@	
1 #include <stdio.h>	1 #include <stdio.h>
2 - void bad_func()	2 + void good_func()
3 {	3 {
4 int a, b, c;	4 int a, b, c;
5	5 +
6 a = 13;	6 a = 13;
7 b = 5;	7 b = 5;
8 - if (c = (a >= b) ? a : b) /* Non-compliant */	8 + c = (a >= b) ? a : b; /* Compliant */
9	9 + if (c != 0)
10 - printf("Bad Case");	10 + printf("Good Case");
11 // only when a == b == 0, nothing happens.	11 // only when a == b == 0, nothing happens.
12 - // else prints "Bad Case"	12 + // else prints "Good Case"
13 }	13 }
14 int main() {	14 int main() {
15 - bad_func();	15 + good_func();
16 return 0;	16 return 0;
17 }	17 }

각 주차별 업무 수행과정 (7주차)

2. Good/Bad Case 설계 및 수정

(Example 3) MISRA_CPP_12_01_03

하나의 기본 타입의 매개변수를 갖는 호출 가능한 생성자들은 explicit으로 선언되어야 한다.

sample CHANGED	
<pre>@@ -1,14 +1,14 @@ 1 class A 2 { 3 public: 4 - A(int a) 5 { 6 var1 = a; 7 var2 = 'A'; 8 } 9 - A(char b) 10 { 11 var2 = b; 12 var1 = 1; 13 } 14 A(int a, char b) @@ -17,6 +17,5 @@ 17 } 18 private: 19 int var1; 20 char var2; 21 };</pre>	<pre>1 class A 2 { 3 public: 4 + explicit A(int a) 5 { 6 var1 = a; 7 var2 = 'A'; 8 } 9 + explicit A(char b) 10 { 11 var2 = b; 12 var1 = 1; 13 } 14 A(int a, char b) 17 } 18 private: 19 int var1; 20 char var2; 21 };</pre>

각 주차별 업무 수행과정 (7주차)

2. Good/Bad Case 설계 및 수정

(Example 4) MISRA_C_2012_08_13

가능하다면 포인터는 const 로 한정된 타입을 가리켜야 함

```
sample [CHANGED]
@@ -1,9 +1,9 @@
1  int value;
2  int value2;
3  - int* func()
4  {
5  -  int* ptr = &value;
6  ...
7  ptr = &value2;
8  return ptr;
9  }
```

```
3  // This is for exception case
4  + // case 1 : replacing ( active situation )
5  int pNum;
6  + void good_func( int* Time ) /* compliant */
7  {
8  +  int* pNum = Time; // exception case - copying address
9  +  *pNum = *pNum + 1;
10 }
```

Update: 2021-0

```
sample [CHANGED]
@@ -1,5 +1,6 @@
1  typedef struct A
2  {
3      int m;
4  }A;
5  - void bad_func ( A* const a ) { } /* Non-compliant */
```

```
1  typedef struct A
2  {
3      int m;
4  }A;
5  typedef A* PtrA;
6  + void bad_func( const A* a ) { } /* compliant */
```

```
sample [CHANGED]
@@ -1,7 +1,10 @@
1  // This is for exception case
2  - // case 1 : being replaced ( passive situation )
3  int pNum;
4  - void bad_func( int* Time ) /* Non-compliant */
5  {
6  -  Time = &(pNum);
7  }
```

```
3  // This is for exception case
4  + // case 2 : replacing ( active situation )
5  int pNum;
6  + void good_func( int* Time ) /* compliant */
7  {
8  +  int* pNum = Time; // exception case - copying address
9  +  *pNum = *pNum + 1;
10 }
```

Update: 2021-0

```
sample [CHANGED]
@@ -1,6 +1,6 @@
1  typedef struct A
2  {
3      int m;
4  }A;
5  typedef A* PtrA;
6  - void bad_func( A* const a ) { } /* Non-compliant */
```

```
1  typedef struct A
2  {
3      int m;
4  }A;
5  typedef A* PtrA;
6  + void bad_func( const A* a ) { } /* compliant */
```

3. Fix Reference 작업

Fix Reference Management

파일 선택

선택된 파일 없음

Import

Export

Management

Service DB

Manage DB

C/C++

JAVA

C#

8069

MISRA_C_2012_08_02

MISRA_CPP_05_00_15

MISRA_CPP_09_03_03

MISRA_C_2012_08_01

MISRA_C_2012_08_13

MISRA_CPP_08_05_01

MISRA_CPP_00_01_07

MISRA_CPP_06_05_01

MISRA_CPP_02_10_03

MISRA_CPP_19_03_01

MISRA_CPP_12_01_03

MISRA_CPP_12_01_02

MISRA_CPP_12_08_01

MISRA_CPP_10_03_01

MISRA_C_2012_17_07

Group ID : MISRA_CPP_06_05_01 (1 개)

Update: 2021-02-19 17:15

MODIFY

DELETE

Home

sample

1 - float32_t bad_1304_var0;

2 - float32_t bad_1304_var1 [3] = { 1.0f, 2.0f, 3.0f };

3 - for (bad_1304_var0 = 0.0f;

4 - bad_1304_var0 < bad_1304_var1;

5 - bad_1304_var0 = bad_1304_var0 + 1.0f)

6 {

7 ...

8 }

9 void func(void){

10 int y = 0;

11 - for(int x = 0; x < y ; x = x++){

12 }

13 }

1 + uint16_t good_1304_var0;

2 + uint16_t good_1304_var1;

3 + for (good_1304_var0 = 0;

4 + good_1304_var0 < good_1304_var1;

5 + good_1304_var0++)

6 {

7 ...

8 }

9 void func(void){

10 int y = 0;

11 + for(int x = 0; x < y ; x++){

12 }

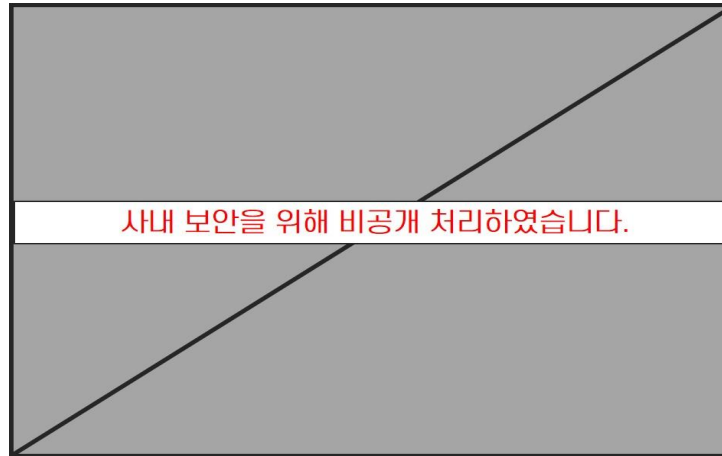
13 }

각 주차별 업무 수행과정 (8주차)

1. 매뉴얼개선 리뷰 회의 참석 및 수정사항 반영



< 매뉴얼 회의 참석 >



< 개선사항 Git 에 반영 >

구분	Good Case 케이스로 등록	Bad Case 케이스로 등록	내용 수정 상황	비고
NISPA_C 2012.08.04	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
NISPA_C 2012.08.04	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
NISPA_C 2012.08.08.1	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
NISPA_C 2012.08.08.2	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
NISPA_C 2012.08.12.1	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
NISPA_C 2012.08.12.2	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
NISPA_C 2012.08.12.3	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
NISPA_C 2012.08.12.4	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
NISPA_C 2012.08.03	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
NISPA_C 2012.12.04	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
NISPA_C 2012.12.05	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
NISPA_C 2012.12.06	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
NISPA_C 2012.12.07	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
NISPA_C 2012.12.08	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
NISPA_C 2012.12.09	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
NISPA_C 2012.12.10	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
NISPA_C 2012.12.11	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
NISPA_C 2012.12.12	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

< 업무에 대한 인수인계 작업>

2. 현장실습 보고서 작성 및 인수인계

- 공유 스프레드시트 소유자 권한 이전
- 수정사항 반영한 문서 정리

업무 성과



사내 보안을 위해 비공개 처리하였습니다.

드린점 및 마무리

느낀점 및 마무리

- 첫 직장생활
- 학부 과목 내용(C / C++) 리뷰 & 응용
- 규칙적인 시간
 - 8시 30분 출근, 5시 30분 퇴근
 - “일찍 일어날 이유” 중 하나가 됨
- 데일리 미팅(Daily Meeting)
 - 입사 초기 팀에서의 업무를 파악하는데 도움이 됨
 - 하루하루 오늘 해야할 일들을 미리 계획 / 실행
- 회의 시간 참여
- 이후의 학습: 컴파일러설계, 소프트웨어공학

Thank you

