

수행내용 보고서

(3 차 보고)

과제명	오프라인 보드게임 “달무리” 의 콘솔 게임화		
발표자	황현석, 이진석	발표일자	2023.06.05
연구참여자 (담당분야)	팀장: 황현석(담당분야: 팀관리, 개발 규칙 정하기, back-end 부분 개발, front-end 부분 오류사항 제시 및 해결, 보고서 작성) 팀원: 이진석(front-end 부분 개발, 보고서 작성), 김민석 (front-end 부분 중 인트로 화면 구현), 이승현(back-end 부분 개발, front-end 부분 오류사항 제시 및 해결, 보고서 작성)		
수업명/조번호	프로그래밍설계/7조		

1. 연구일정 점검

1) 단기일정 : 2023년 5월 09일 ~ 2023년 6월 05일 (이전, 이후 단계는 절사)

수행 내용	일정 (3~4일 단위)								달성 실적
	1	2	3	4	5	6	7	8	
- 협업을 위한 코드 작성 규칙 정하기									완료
- 프로그램 로직 부분 코딩									90% 달성
- UI 구현									70% 달성

위의 달성 실적란에는 발표일 기준으로 한 학기 기간을 고려하여 전체 목표 대비 발표일 시점의 실적 비율을 표시하며 완료한 수행내용은 완료로 표시함.

2. 연구수행 내용

: 지난 발표 후 연구수행 내용을 구체적으로 적을 것

: 제출일 기준 발표자료 내용과 설명을 포함할 것

[Back-end]

1. 변수/함수명 통일

처음 전체적인 큰 함수 4개를 4명의 팀원이 각각 1개씩 담당하여 따로 작업을 하였기에 모든 함수를 다시 하나의 프로그램으로 합치는데 생각보다 많은 시간이 들었습니다. 그 이유중 하나가 서로의 함수/변수명이 통일 되지 않고 의미 파악도 쉽게 되지 않았기 때문에 이번주 작업 중 가장 먼저 해야 할 일이라고 생각했습니다. 우선 기존 card_count, card_name 같은 변수명을 cardCount, cardName 등 전체적으로 모든 변수명은 카멜표기법으로 변경했습니다. 또한 의미 파악이 한눈에 되지 않는 변수/함수명들을 의미 파악이 쉽게 변경했습니다. 예를들면 pass_3 -> passCount, now_player_card() -> delete_submitted_card(), cards[20] -> subCardArr[20] 등으로 한눈에 이 변수/함수가 어떤 역할을 하는지 파악하기 쉽게 변수/함수명을 바꿨습니다.

2. goto문 삭제

기존 코드에 있었던 goto문을 삭제하였습니다. 간단한 코드에서는 문제가

없을 수 있으나 최종 프로젝트까지 코드길이가 길어지고 복잡해짐에 따라 goto문은 코드 진행을 꼬이게 할 수 있는 매우 위험한 요소이기 때문에 이번 회의에서 빼기로 했습니다. 원본 코드의 goto문은 잘못된 카드를 제출하였을 때(ex. 카드 번호 범위 위반, 자신의 패에 없는 카드 제출 등) 다시 카드를 재제출하기 위해 사용되었습니다. 이것을 While 반복문으로 바꿔 카드를 정상 제출하거나 pass를 하였을 조건에 조건문을 빠져나오고 그 이외에 모든 오류 상황은 카드 제출 로직을 반복하게 만들었습니다.

3. 전역변수 사용

여러 가지 함수를 합치는 과정에서 다른 소스파일에 같은 역할을 하는 변수를 불러오고 싶을 때 사용을 용의하게 하기 위한 변수들은 전부 전역변수 처리를 했습니다. 다른 소스파일에서도 extern 식별자를 통해 동일한 변수를 사용하고 가져올 수 있게 했습니다. 또한 다른 소스파일 뿐만 아니라 하나의 소스파일 안에서도 여러 가지 사용자지정함수에 반복적으로 나오는 변수들은 전부 전역변수 처리를 하여 코드 관리를 조금더 용의하게 하였습니다.

4. 턴 진행방식 변경

이전 진행 상황에서는 턴 진행을 Queue자료 구조를 도입하기로 계획했으나 실제 프로젝트를 구성하는 단계에서 코드가 불필요하게 복잡해지는 것을 느껴 변경하였습니다. 플레이어의 패를 저장하는 2차원 정수배열 playerCards와 플레이어들의 닉네임을 저장하는 2차원 문자배열 player_name의 인덱스를 활용하여 간단하게 턴을 돌아가는 로직을 새로 구현했습니다.

5. 카드 제출 방식 변경

기존 프로그램에서는 카드를 제출 할 때 '12/3' 이런식으로 scanf에 인수를 입력받아 12번 카드 3장 제출이라는 의미를 활용했지만 사용자가 입력할 때 식별이 어렵거나 복잡할수도 있다고 생각했습니다. 개발 초기 단계에도 카드 제출 방식은 '12 12', '11 11 11 11'등 원하는 카드를 원하는 장수만큼 직접 타이핑하는 것을 원했습니다. 그럼에도 이전에 저 방법을 사용한 이유가 12를 2장 제출한다면 scanf에 2개의 인자를 받아야하고 11을 4장 제출한다면 인자를 4개를 받아야 하는 등 매입력마다 인자의 개수가 미지수이기 때문에 이 문제점을 해결하지 못했습니다. 하지만 이번주 회의 결과 gets_s

함수에 원하는 카드번호와 장수를 문자열 배열로 받은 다음에 문자열 자르기 함수인 strtok()함수 활용하여 ‘공백’을 기준으로 문자열을 잘라 사용자가 입력한 카드 번호를 알고 전역변수 subCardArr 배열에 저장하고 잘린 개수를 이용하여 카드 장수를 return 받을 수 있게 만들었습니다.

6. 상세 룰 추가

이전 진행 상황에서는 상세적인 룰까지 카드 제출 로직에 도입하지는 못했습니다. 그 중 프로그램에 도입하지 못한 룰 중 하나인 이전 플레이어보다 ‘높은 등급’의 카드를 ‘같은 장수’ 만큼 제출 알고리즘을 도입하지는 않았습니다. 그래서 이번주에 이것도 도입하였습니다. 이전 플레이어가 제출한 카드 번호와 카드 장수를 각각 beforeCardName고 beforeCardCount 전역변수에 저장을 한 뒤 이후 플레이어가 제출한 카드가 beforeCardName보다 낮거나(이전 플레이어보다 높은 등급의 카드 제출) beforeCardCount와 같지(이전 플레이어와 같은 장수의 카드 제출) 않다면 오류 코드가 생기게 했습니다.

또한 위에 로직을 추가함에 따라 턴 초기화 조건(마지막으로 제출한 플레이어 이외에 모든 플레이어가 pass를 할시 처음부터 원하는 카드를 원하는 장수만큼 제출할 수 있도록 초기화)도 추가했어야 했습니다. 이를 위해 전역변수 passCount변수를 만들어 플레이어가 카드를 제출하였다면 passCount를 0으로 초기화 하고 플레이어가 pass를 하였다면 passCount를 1씩 증가시켜 4인 플레이 기준 passCount >= 3 조건을 만족하였을 때 beforeCardName과 beforeCardCount 변수를 초기화 하여 다음 플레이어는 카드 제출에 제약 없이 원하는 카드를 원하는 장수만큼 제출 할수 있도록 기능을 추가했습니다.

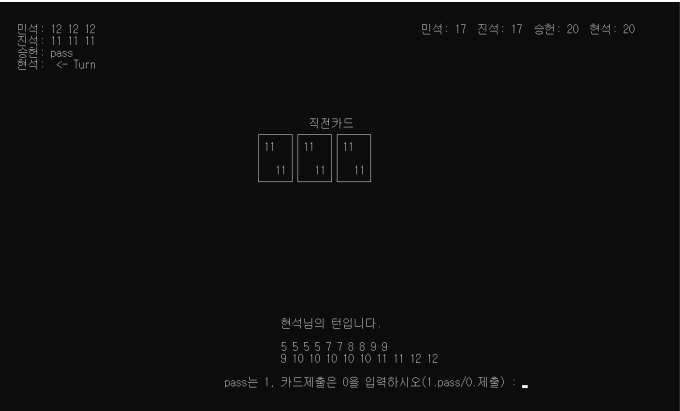
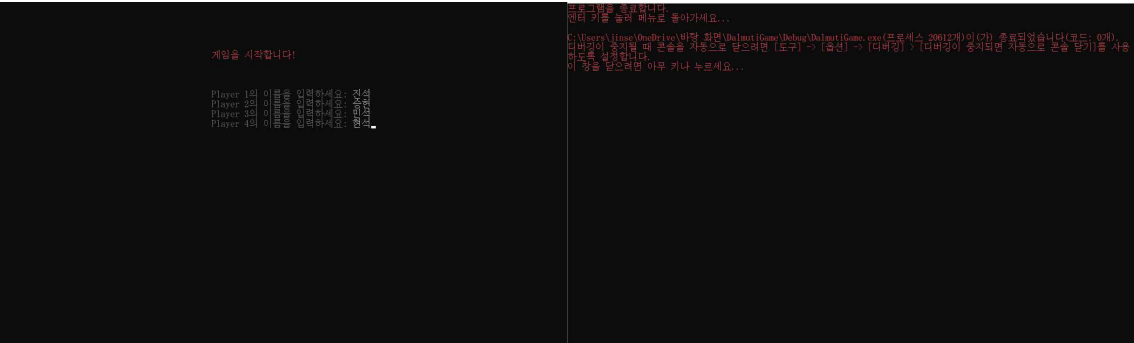
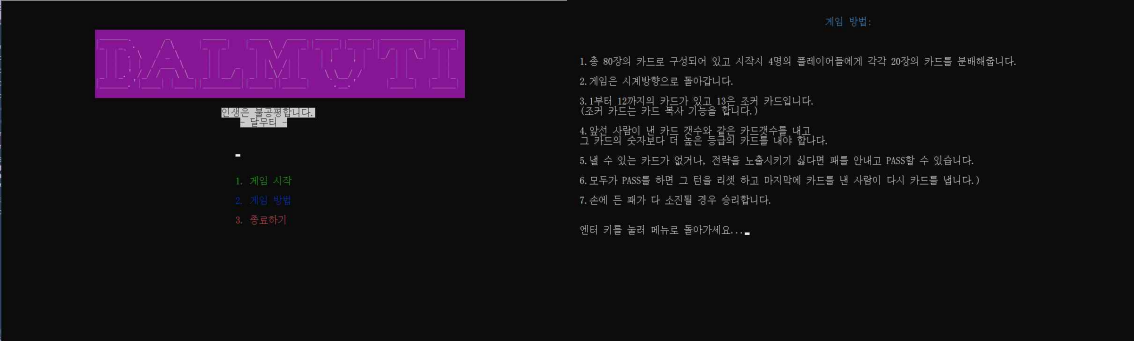
[front-end]

1. 콘솔 화면 구성하는 코드 작성 (SceneManager)

- mainMenu: 실행시 처음 보이는 타이틀화면입니다.
- showGameRules: 게임의 룰을 설명하는 페이지로 메뉴에서 접근 가능합니다
- setPlayersName: 게임시작 시 플레이어의 이름을 정하는 페이지입니다
- mainGameScene: 게임 메인화면입니다.

gotoxy와 printf 함수를 활용해서 화면을 구성하였고, 글자에 색을 입히는 작업을 하여 꾸며주었습니다. 해당 위치에 전역 변수로 할당한 변수값을 받

아와 출력하는 형식으로 구현하였습니다.



3. 연구계획 (향후 연구수행 예정인 내용 요약)

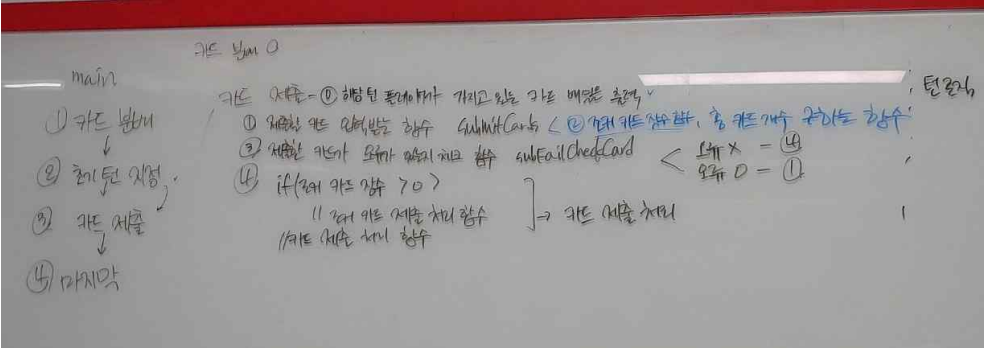
구분		수행내용 요약
팀전체		달무티 게임의 카드 로직을 함수별로 구분 한 후 개별 구현하여 헤더 파일로 합치기.
개인별	황현석	헤더 파일과 소스 파일(.cpp)파일 분리 및 코드 작성 요령을 팀원에게 교육, 코드 작성 규칙 및 코드 관리 교육(변수명/함수명, 깃허브 리포지토리 사용 방법, 테스트 케이스 사용 방법) back-end와 관련된 기능 총괄, 구현 Front-end 코드 테스트 및 오류 수정 각 팀원에게 역할 지시
	이진석	Front-end 총괄, Front-end 기능 구현 전체 코드파일 취합
	김민석	Front-end 게임 초기 화면 기능 구현 테스트 케이스 제작
	이승헌	back-end와 관련된 기능 구현 Front-end 코드 테스트 및 오류 수정

4. 문제점 및 애로사항

: 과제 진행중에 발생된 문제점들과 이를 해결한 내용들

: 애로 사항들을 구체적으로 기술한다.

회 의 록			
회의명	Back-end 최종 로직 구현		
일 시	23. 05. 25	장 소	학술정보관 + 성결관 1층 VR센터장실
의 제	Back-end 최종 로직 구현을 위한 설계 및 구현		
회의 내용	1. Back-end 로직 구현을 위한 설계 논의 2. 달무티 룰 통일 점검		

결의 사항	<p>1) 카드 분배 로직 수정 카드 정렬, 배분 받은 카드를 화면에 출력 부분은 front-end 부분으로 옮기고 카드 분배 로직은 카드 분배 기능만을 수행하도록 한다.</p> <p>2) 게임 턴 관리 로직 구현 방식 설정 기존 '원형 큐 방식'을 사용할 경우 정보통신공학과 2학년 학부생 수준에서 막 큐를 배우고 있는 단계라 원형 큐를 적용하기 어려움이 있다고 판단하여 이를 배려해 '배열을 통한 인덱스 접근' 방식을 사용한다. 성능도 크게 차이가 없고, 더 코드 해석이 쉬워 이 방식을 채택하게 되었다.</p> <p>3) goto문에서 while문으로 전환 goto문은 스파게티코드를 유발한다. while문으로 수정하여 구현한다.</p> <p>4) 룰 통일 작업 같은 게임이더라도 다른 룰을 생각하고 구현하면 안되므로, 확실하게 통일하고 구현에 넘어가야 한다.</p> 			
이건 사항	없음			
참석 현황	대상	성명	학번	서명(자필)
	참석자	<p>황현석</p> <p>이승현</p>	<p>20191033</p> <p>20200931</p>	

				<div> <div>황현석</div> <div>이진석</div> <div>김민석</div> <div>이승현</div> </div>
	불참자	이진석 김민석	20191008 20200892	
불참 사유	Back-end 회의라 Front-end는 참여할 필요성이 없었음.			
작성자	황현석	작성일	23. 05. 25	

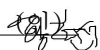



회 의 록				
회의명	front-end 최종 로직 구성 및 코드 통합			
일 시	23. 05.21	장 소	학술정보관 성결관 1층	
의 제	프론트 엔드 및 전체 코드 흐름 관리			
회의 내용	프로그램 실행 시 초기 화면 구성 로직 구상 전체 코드 통합 관리			
결의 사항	화면 시작 시 콘솔창에 게임 이름'DALMUTI' 출력되게 함 및 콘솔창 화면 출력 시 gotoxy로 위치 조정해서 출력함과 동시에 1.게임 시작 2.게임 방법, 3.종료하기 색깔을 다르게 표시 및 해당하는 로직 구현. 백엔드 구현 완료시 전체 코드 통합해서 실행시키기.			
이견 사항	없음			
참석 현황	대상	성명	학번	서 명(자필)
	참석자	이진석 김민석	20191008 20200892	황현석 이진석 김민석 이승헌
	불참자	황현석 이승헌	20191033 20200931	
불참 사유	Back-end 부분은 참여할 필요성이 없었음.			
작성자	김민석	작성일	23. 05. 21	

회 의 록			
회의명	코드 작성 규칙 통일화		
일 시	23. 05. 15	장 소	VR센터장실
의 제	협업을 위한 코드 작성 규칙 통일화 교육		
회의 내용	1) 변수명/함수명 논의 2) 헤더 파일과 소스 파일의 분리 방법 교육 3) 깃허브 리포지토리 공유 및 리포지토리 사용 방법 교육		
결의 사항	<p>- 변수명/함수명 규칙</p> 1. 가급적 상수는 전처리기로 정의하고 상수는 대문자로 사용한다. 2. 길이가 길어도 되니 직관적인 변수/함수명을 사용한다. 3. 카멜 표기법을 사용한다. 4. 관례적인 변수인 경우 해당 변수를 사용한다. (for문의 I변수) 5. 한 가지 함수는 한 가지의 일만 하도록 정의한다. <p>- 헤더 파일과 소스 파일의 분리 방법</p> 1. 헤더 파일은 함수의 선언만 한다. 2. 소스 파일은 함수의 정의를 한다. 3. 소스 파일에서 다른 소스 파일이 필요한 경우 “ ” 로 해당 헤더를 include 한다. <p>- 깃허브 리포지토리 교육</p> 공유한 리포지토리 => 아래 링크 참고 https://github.com/HwangHyeonseok/DalmutiGame 자신이 수정하거나 만든 소스파일 올리는 방법? Add file -> Create new file		
이견 사항	없음.		

참석 현황	대상	성명	학번	서명(자필)
	참석자	황현석 이진석 김민석 이승현	20191033 20191008 20200892 20200931	황현석 이진석 김민석 이승현
	불참자			
불참 사유				
작성자	황현석, 이진석	작성일	23. 05. 15	

5. 지도교수 코멘트 내용 : 지도교수님한테 상담받은 내용 또는 수업시간에 받은 코멘트를 정리함

지도교수 상담일지

과제명	오프라인 보드게임 “달무티”의 콘솔 게임화		
일지 NO	2	학년도/학기	2023 학년 / 1 학기
소속	미디어소프트웨어학과 3학년 / 정보통신공학과 2학년		
지도교수 확인	2021년 5월 15일	지도교수:	(인)
상담 안건	1. 5월 15일 발표 피드백과 앞으로의 일정 (룰 설명이 이해되기 쉽게 청중의 입장에서 적절한 설명이었던지 의문점 해결, 최종보고 발표때 시연은 어떤 식으로 보여주면 되는지에 대한 의문점 해결) 2. 오버로딩으로 여러 개의 코드(12개)를 작성할 경우 코드가 더러워지지 않는지에 관한 질문 3. [팀장 개인 상담 질문] 학생 연구원에 관한 상담		
상담 내용	<p>Q. 5월 15일, 해당 발표가 달무티 룰을 청중에게 이해시키는데 적절했는지와 앞으로도 이런 룰 설명을 지속적으로 할 필요가 있을지에 대한 의문점</p> <p>A. ppt 애니메이션을 활용하여 저번보다 달무티 룰을 청중에게 이해시키는데는 도움이 많이 되었던 것 같다.</p> <p>하지만 일주일 뒤면 룰을 또 까먹는 경우가 많으므로 룰 설명을 지금처럼 초반에 간단하게 설명하고 넘어가는 것이 청중의 이해와 흥미를 높여주는데 좋을 것 같다.</p> <p>반영 : 해당 슬라이드를 그대로 청중들에게 3차, 4차 발표때도 설명을 하여 달무티 룰을 지속적으로 상기할 수 있도록 한다.</p> <p>Q. scanf 입력 받는 인자의 경우의 수가 12개여서 오버로딩을 통해 구현을 하고자 하는데 오버로딩을 통해 이렇게 여러개의 함수를 만들 경우 코드를 더러운 코드라고 판정하시지는 않을까요?</p> <p>A. 모듈화를 한 코드를 더러운 코드라고 판단할 수는 없다. 12개의 함수를 오버로딩으로 구현해도 된다.</p> <p>Q. 변수명, 함수명을 지을 때 카멜 표기법, 스네이크 표기법은 크게 상관이 없는지 궁금합니다.</p> <p>A. 크게 상관 없다. 통일성만 있으면 된다.</p>		
코멘트에 대한 조치사항	이번 상담을 통해 오히려 함수를 최대한 많이 사용하고 main 부분을 짧게 가져가는 것이 더 좋다고 판단하여 최대한 함수를 많이 사용하고 main 부분을 짧게 가져가고자 설계하였고, 협업을 위해 변수명 규칙을 팀 내에서 정하여 사용하였으며 .cpp 파일과 .h(헤더 파일)을 분리하여 팀 작업이 원활하게 되도록 개발 환경을 조성하였다.		
상담 참여자	<div style="display: flex; justify-content: space-between;"> <div> <u>황현석</u>  <u>이승현</u>  </div> <div> <u>이진석</u>  <u>김민석</u>  </div> </div>		