2주차 예비보고서

전공: 컴퓨터공학과 학년: 2학년 학번: 20211558 이름: 윤준서

**1. 목적**

UNIX 상에서 제공하는 C/C++관련 도구를 미리 사용해 봄으로써, 수업시간에 실습이 원활히 진행될 수 있도록 한다.

**2. 예비 학습**

C/C++ 프로그램의 컴파일 과정으로 프리프로세싱(Preprocessing) 단계, 컴파일링(Compiling) 단계, 어셈블링(Assembling) 단계, 그리고 링킹(Linking) 단계의 총 네 단계로 구성되어있다.

**1. 프리프로세싱(Preprocessing) 단계**

'전처리기' 라고도 불리우는 프리프로세싱 단계는 특정 코드의 변경과 추가를 통해 프로그래밍 언어를 확장시키는 역할을 한다. #include 을 통해 헤더 파일을 포함하거나 #define 을 이용해 매크로를 생성하는 등의 기능이 있다. 특정 코드 사용의 예시는 다음과 같다.

#include <stdio.h> C/C++ 언어 함수의 매크로 정의, 여러 입출력 함수 등의 기능이 포함된 헤더 파일이다.

#include <string.h> C/C++ 언어 함수의 문자열과 관련된 함수의 기능이 포함된 헤더 파일이다.

#definde max\_number 100 매크로를 통한 변수의 값 고정을 한다. 이를 통해 프 로그램에서 max\_number 가 나올 때마다 100의 값으 로 바꾼다.

**2. 컴파일링(Compiling) 단계**

프리프로세싱 단계를 포함한 C/C++ 코드 언어를 어셈블리어로 변환한다. 컴파일링 단계는 프론트엔드(front-end), 미들엔드(middle-end), 백엔드(back-end) 의 세 단계로 나뉜다.

1. 프론트엔드(front-end)

코드의 문법과 어휘가 올바른지 확인하고 코드를 트리 형태로 표현한 자료 구조인 GIMPLE 트리를 생성하여 미들엔드에 넘긴다.

2. 미들엔드(middle-end)

받은 GIMPLE 트리를 통해 비종속적 최적화를 시행 후, 코드 언어와 어셈블리어의 중간 형태의 RTL(Register Transfer Language) 을 생성하여 백엔드에 넘긴다.

3. 백엔드(back-end)

받은 RTL 을 통해 종속적 최적화를 시행 후, 어셈블리 코드를 생성한다.

**3. 어셈블링(Assembling) 단계**

기계어 번역 프로그램인 어셈블러를 이용해 컴파일 단계에서 얻은 어셈블리 코드들을 호출, 해석하여 기계어로 번역한다. 이후 오브젝트 코드(목적 코드)를 생성한다.

**4. 링킹(Linking) 단계**

어셈블링 단계에서 생성된 오브젝트 코드들을 결합하여 오브젝트 파일을 생성한다. 이를 이용해 실행이 가능한 실행 파일을 생성한다. 리눅스의 경우 a.out 파일이 그 기본이지만 gcc -o 를 이용해 실행 파일의 이름을 바꿀 수 있다.