**Pintos Project 1: User Program (1)**

담당 교수 : 김영재

조 / 조원 : 20171663 이도훈

개발 기간 : 2021.09.10 – 2021.10.02

1. **개발 목표**

* Pintos의 기본 코드를 이용하여 userprogram을 구현한다. Argument들을 tokenize 시켜서 패싱하고 wait, exit, exec, halt 등의 다양한 system call과 추가적으로 fibonacci와 4개의 정수 중 최대 값을 구하는 max\_of\_four\_int도 함께 구현했다

1. **개발 범위 및 내용**
   1. **개발 범위**
2. Argument Passing

입력받은 argument가 tokenized 되어 스페이스바를 기준으로 나뉜 뒤 스택에 차례대로 저장된다.

1. User Memory Access

메모리에 접근하는 과정에서 주소가 잘못 넘겨지게 되면 예외 처리를 통해 프로그램을 종료한다.

1. System Calls

스택에 있는 명령어를 system call handler를 통해 적절히 처리된다.

* 1. **개발 내용**
* Argument Passing

먼저 입력받은 명령어를 tokenize 시키고 분리된 명령어들은 esp라는 스택 포인터를 사용하여 stack에 차례대로 저장된다. 명령어가 모두 저장이 된 이후에는 word\_align을 스택에 저장하고, 저장된 스택의 명령어의 address를 스택에 저장하고, 명령어의 개수, return address까지 스택에 저장한다.

* User Memory Access

적합하지 않은 메모리 접근을 막는다. 적합하지 않은 메모리 접근에 해당하는 예시로는 NULL 포인터와 Unmapped virtual memory, 그리고 kernel address space의 pointer가 주소공간에 access할 때를 말한다. User process는 user memory 공간만 사용할 수 있도록 kernerl space가 접근하는 것을 막아야 한다.

따라서 User process가 address space에 접근하기 전 여러 검사를 통해 kernerl address의 포인터인지 확인해야 한다. 이러한 과정을 통해 적합하지 않은 메모리 접근을 방지한다.

* System Calls

운영체제는 kernel과 user로 나뉘어 있다. Kernel은 파일을 읽거나 메시지를 출력할 때 많이 사용된다. System Call은 이러한 kernel 영역의 기능을 user 모드에서 사용가능하게 한다. 이번 프로젝트에서는 Halt, exit, exec, wait 등을 구현했다.

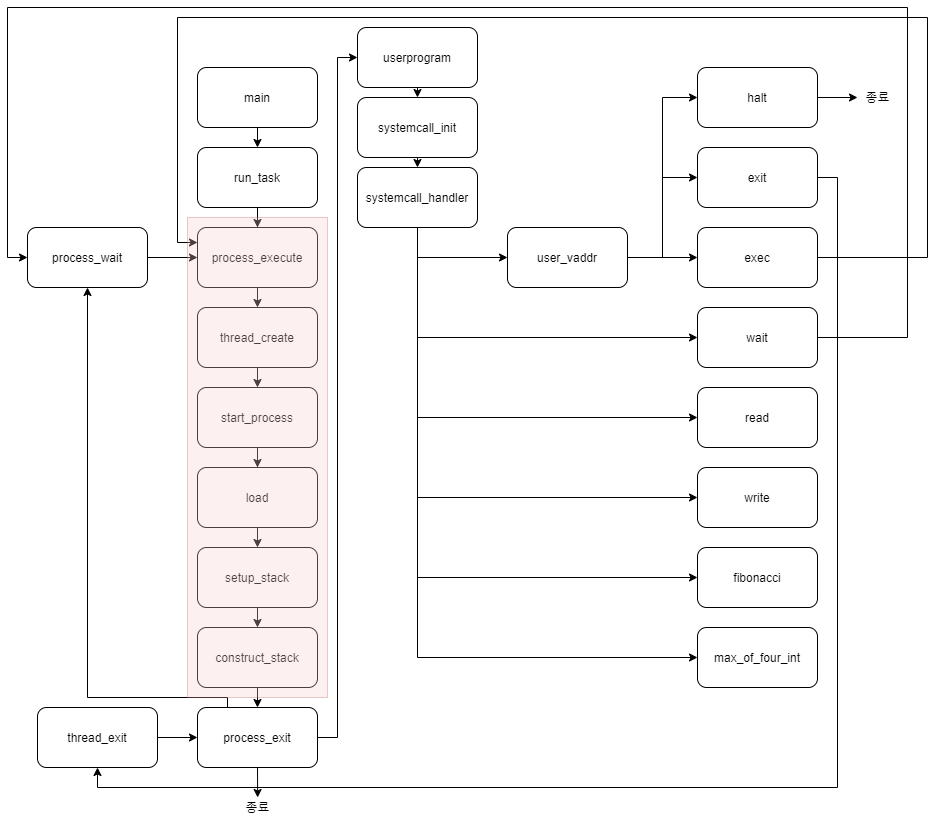
1. **추진 일정 및 개발 방법**
   1. **추진 일정**

09.20 – 09.30 argument passing, system call user memory access 구현

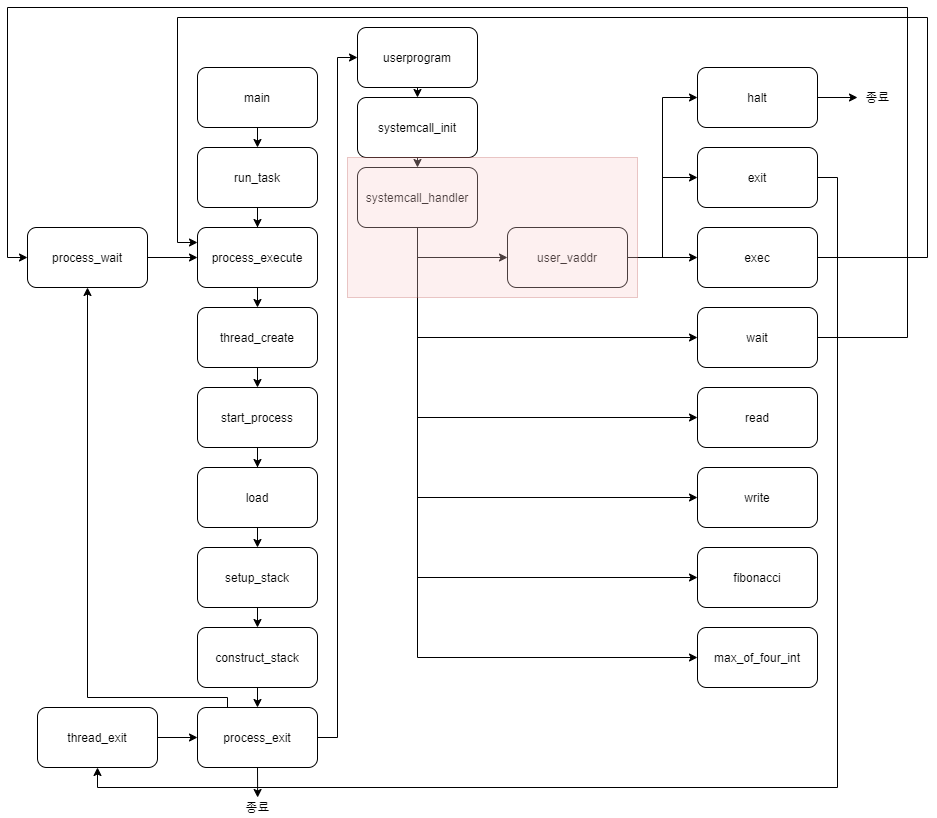
09.30 – 10.02 additional 구현

10.02 – 10.02 보고서 작성

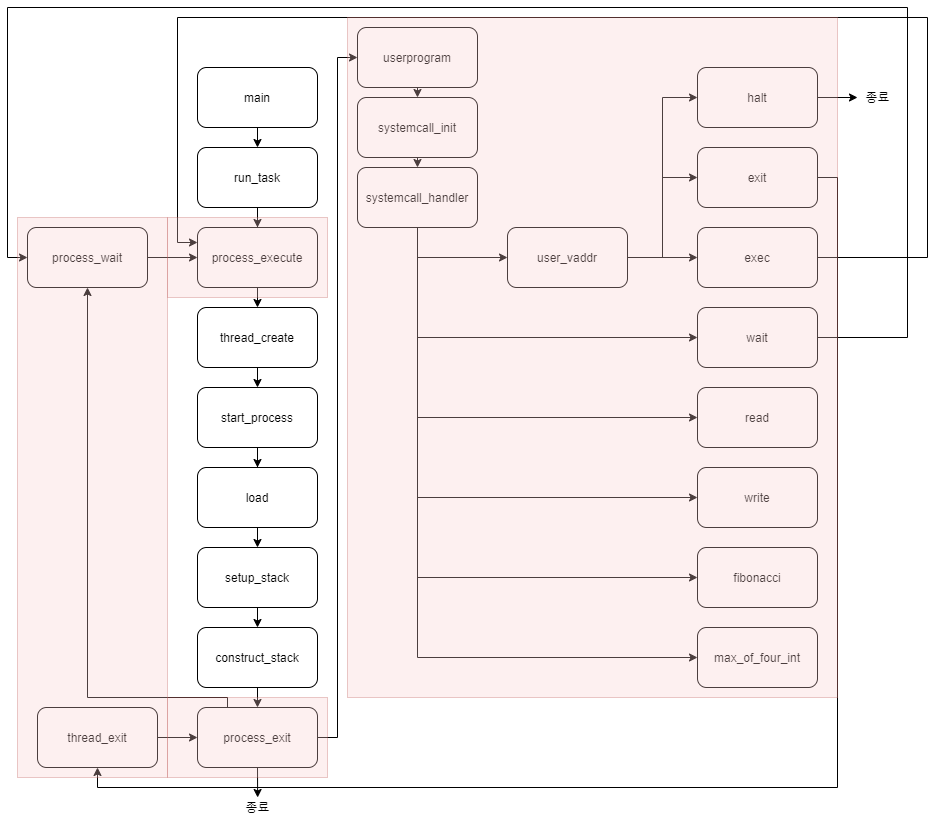
1. **연구 결과**
   1. **Flow Chart**
2. Argument Passing



1. User Memory Access



1. System Calls



* 1. **시험 및 평가 내용**
* **텍스트이(가) 표시된 사진

  자동 생성된 설명**