1. **파이썬의 라이브러리인 배열을 이용해 LRU Simulator 코드를 완성하시오.   
   템플릿 코드:** [**https://github.com/eunjicious/ds\_2024.git**](https://github.com/eunjicious/ds_2024.git)

소스코드

class CacheSimulator:

    def \_\_init\_\_(self, cache\_slots):

        self.cache\_slots = cache\_slots

        self.cache\_hit = 0

        self.tot\_cnt = 0

        self.cache = []

    def do\_sim(self, page):

        self.tot\_cnt += 1

        if page in self.cache:

            # 페이지가 캐시에 이미 있으면, 캐시 히트이며, 해당 페이지를 캐시의 끝으로 이동 -> python 내부에서 append 를 어떻게 최적화 할지는 모르지만 만약 [] 가 C++ 기준 배열이라면, 비효율적인 동작, 만약 연결 리스트 방식이면, 그나마 효율이 있다

            self.cache\_hit += 1

            self.cache.append(self.cache.pop(self.cache.index(page)))

        else:

            # 페이지가 캐시에 없으면, 새로운 페이지를 추가

            if len(self.cache) >= self.cache\_slots:

                # 캐시가 가득 차 있으면, 가장 오래된 페이지(리스트의 첫 번째 요소)를 제거 -> 이 역시 동일하게 효율이 아쉽다.(내부 구현을 모르기에 어떻게 작동하는지를 모르겠음)

                self.cache.pop(0)

            self.cache.append(page)

    def print\_stats(self):

        print("cache\_slot = ", self.cache\_slots, "cache\_hit = ", self.cache\_hit, "hit ratio = ", self.cache\_hit / self.tot\_cnt)

if \_\_name\_\_ == "\_\_main\_\_":

    data\_file = open("C:/Users/me/Desktop/ssu/2-1/DS/Week4/report/1/linkbench.trc")

    lines = data\_file.readlines()

    for cache\_slots in range(100, 1001, 100):

        cache\_sim = CacheSimulator(cache\_slots)

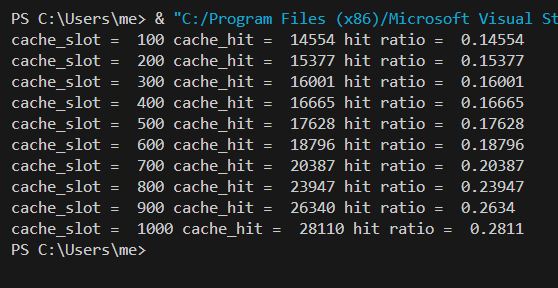
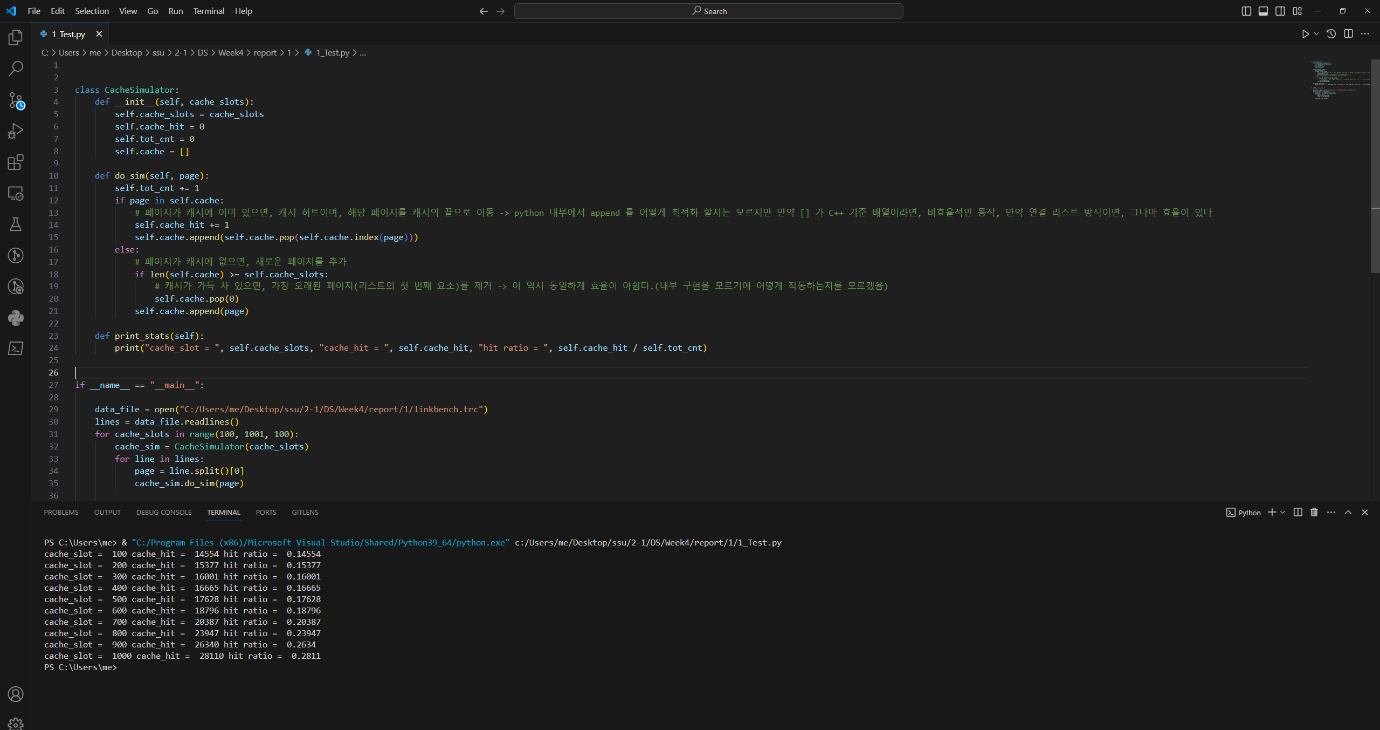
        for line in lines:

            page = line.split()[0]

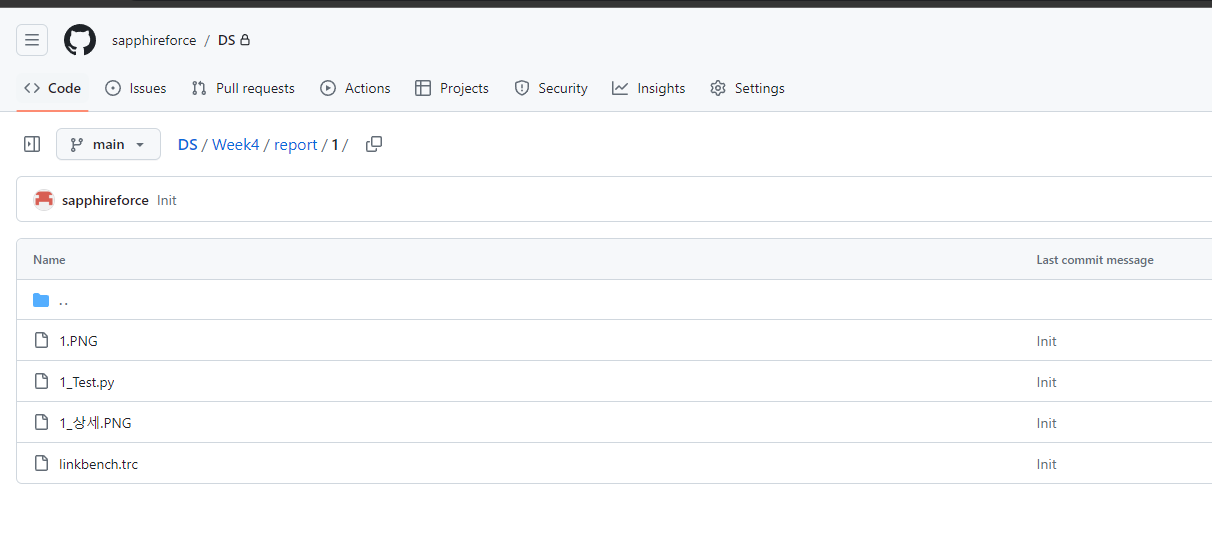
            cache\_sim.do\_sim(page)

        cache\_sim.print\_stats()

결과물



Git push 화면



간단한 설명

가장 단순한 방식으로 밀어넣고 삭제하고 옮기기로 구현했습니다. 속도가 빠를 것 같지는 않을 것 같습니다.

1. **파이썬의 연결리스트 (LinkedListBasic 에 tail 을 추가한 버전 또는 CircularLinkedList) 을 이용해 LRU Simulator 코드를 완성하시오.**

소스코드

class Node:

    def \_\_init\_\_(self, value):

        self.value = value

        self.next = None

class CircularLinkedList:

    def \_\_init\_\_(self):

        self.tail = None

        self.index = {}

    def append(self, value):

        if value in self.index:

            self.move\_to\_end(value)#끝으로 이동

        else:

            NewNode = Node(value)

            if not self.tail: #없으면 새로 만들어주기

                self.tail = NewNode

                self.tail.next = NewNode

            else: #아니면 꼬리 물어주기

                NewNode.next = self.tail.next

                self.tail.next = NewNode

                self.tail = NewNode

            self.index[value] = NewNode

    def remove\_Oldest(self):

        if not self.tail:

            return None

        Oldest = self.tail.next

        if Oldest == self.tail:

            self.tail = None

        else:

            self.tail.next = Oldest.next

        del self.index[Oldest.value] #지우기(중요 -> 버그 발생 지역 예상)

        return Oldest.value

    def move\_to\_end(self, value): #버그 다발 지역(ㅎㅎㅎ)

        if value in self.index and self.index[value] != self.tail:

            node = self.index[value]

            if node.next == self.tail.next:

                self.tail = node

            else:

                prev\_node = self.\_get\_prev\_node(node)

                prev\_node.next = node.next

                node.next = self.tail.next

                self.tail.next = node

                self.tail = node

#------------------------------------------------------------#

    def \_get\_prev\_node(self, node):

        current = self.tail

        while current and current.next != node:

            current = current.next

        return current

#------------------------------------------------------------#

class CacheSimulator:

    def \_\_init\_\_(self, cache\_slots):

        self.cache\_slots = cache\_slots

        self.cache\_hit = 0

        self.tot\_cnt = 0

        self.cache = CircularLinkedList() #왜 안되는겨? -> 한참을 찾았네

    def do\_sim(self, page):

        self.tot\_cnt += 1

        if page in self.cache.index:

            self.cache\_hit += 1

        else:

            if len(self.cache.index) >= self.cache\_slots:

                self.cache.remove\_Oldest()

        self.cache.append(page)

    def print\_stats(self):

        hit\_ratio = self.cache\_hit / self.tot\_cnt if self.tot\_cnt > 0 else 0

        print(f"cache\_slot = {self.cache\_slots}, cache\_hit = {self.cache\_hit}, hit ratio = {hit\_ratio:.4f}")

if \_\_name\_\_ == "\_\_main\_\_":

    data\_file = open("C:/Users/me/Desktop/ssu/2-1/DS/Week4/report/2/lru\_sim/linkbench.trc")

    lines = data\_file.readlines()

    for cache\_slots in range(100, 1001, 100):

        cache\_sim = CacheSimulator(cache\_slots)

        for line in lines:

            page = line.split()[0]

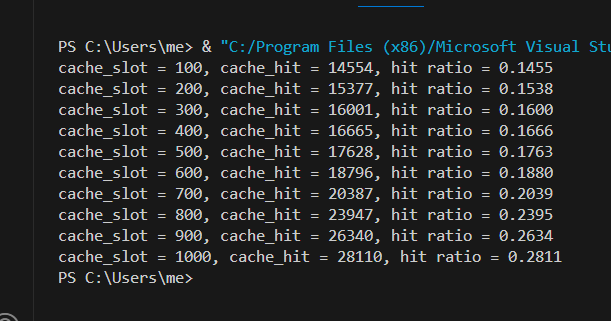
            cache\_sim.do\_sim(page)

        cache\_sim.print\_stats()

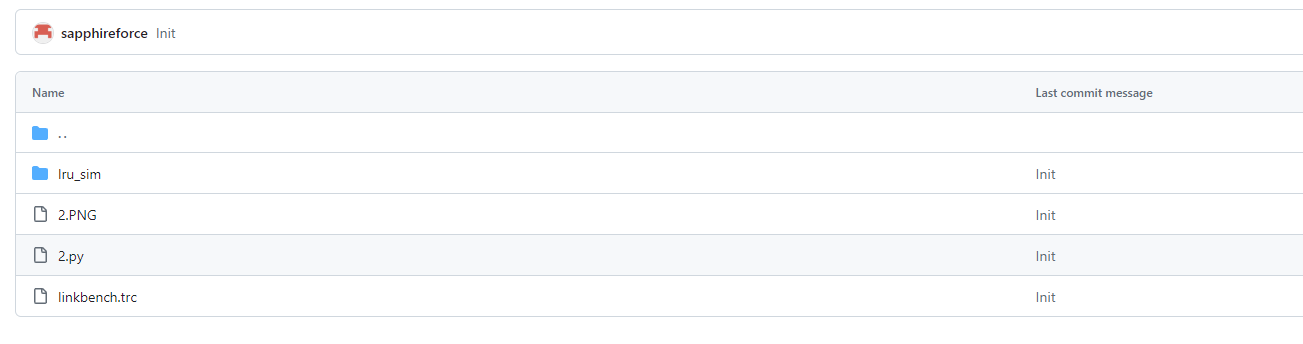
#근데 hit rate가 이정도 나오는게 맞나? 모르겠네

결과물텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명



Git push 화면



간단한 설명

전에 작성한 코드를 최대한 참조해서 만들어 봤습니다.

뭔가 빼먹은 버그나 이상한 작동이 분명히 있을 것 같은데 안보여서 조금은 걱정됩니다.