|  |
| --- |
| **[511643] 자료구조** |
| **실습 #07 보고서** |

|  |  |
| --- | --- |
| **이름** | 곽영주 |
| **학번** | 20175105 |
| **소속**  **학과/대학** | 빅데이터 |
| **분반** | 03 (담당교수: 김태운) |

## <주의사항>

* 개별 과제 입니다. (팀으로 진행하는 과제가 아니며, 모든 학생이 보고서를 제출해야 함)
* 각각의 문제 바로 아래에 답을 작성 후 제출해 주세요.
  + 소스코드/스크립트 등을 작성 한 경우, 해당 파일의 이름도 적어주세요.
* 스마트캠퍼스 제출
  + 데드라인: ~~2020.04.29. ~ 2020.05.05. (화) 23:59~~   
    => 2020.05.01. ~ 2020.05.07. (목) 23:59
    - 과제 업로드가 늦어진 관계로, 과제 제출 일정이 변경되었습니다.
  + 데드라인을 지나서 제출하면 24시간 단위로 20%씩 감점(5일 경과 시 0점)
  + 주말/휴일/학교행사 등 모든 날짜 카운트 함
  + 부정행위 적발 시, 원본(보여준 사람)과 복사본(베낀 사람) 모두 0점 처리함
  + 예외 없음
* 스마트캠퍼스에 아래의 파일을 제출 해 주세요
  + 보고서(PDF 파일로 변환 후 제출)
  + 보고서 파일명에 이름과 학번을 입력 해 주세요.
  + 소스코드, 스크립트, Makefile 등을 작성해야 하는 경우, 모든 파일 제출 (미 제출시 감점)

## <개요>

이번 과제는 큐를 구현하고 활용하는 내용입니다.

## <실습 과제>

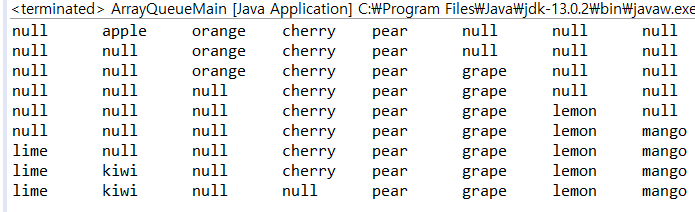
|  |
| --- |
| **[Q 0] 요약 [배점: 10]**  이번 과제에서 배운 내용 또는 과제 완성을 위해서 무엇을 했는지 2~3문장으로 요약하세요. |

답변: 과제를 완성하기 위해 교수님께 이메일을 보냈던 것이 도움이 되었습니다. 그리고 몇 주 동안 동적배열과 리스트를 이용하여 과제를 풀면서 자연스레 복습이 되니 수월하게 과제를 완성할 수 있었습니다.

|  |
| --- |
| **[Q 1] 배열을 이용한 큐 [배점: 15]**  강의자료와 동일하게 ArrayQueue을 구현하세요. ArrayQueue을 테스트 하기 위한 main 함수도 강의자료와 동일하게 구현하고 실행하세요. 터미널 출력 결과를 캡처하여 본 문서에 첨부하세요. 소스코드도 첨부파일로 제출해야 합니다. |

답변:

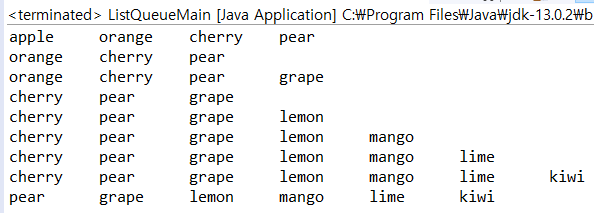
소스코드 : [Q1] 배열을 이용한 큐 소스코드.txt



|  |
| --- |
| **[Q 2] 단순 연결 리스트를 이용한 큐 [배점: 15]**  강의자료와 동일하게 ListQueue을 구현하세요. ListQueue을 테스트 하기 위한 main 함수도 강의자료와 동일하게 구현하고 실행하세요. 터미널 출력 결과를 캡처하여 본 문서에 첨부하세요. 소스코드도 첨부파일로 제출해야 합니다. |

답변:

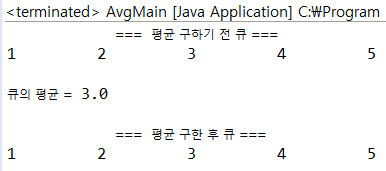
소스코드 : [Q2] 단순 연결 리스트를 이용한 큐 소스코드.txt



|  |
| --- |
| **[Q 3] 평균 구하기 [배점: 20]**  ArrayQueue 또는 ListQueue를 사용해서, 큐에 저장된 int형 숫자의 평균을 구하는 public double avg() 메소드를 구현하세요. add/remove 연산만 사용해서 N개 정수의 평균을 구해야 합니다. 큐에는 int 형의 정수가 N개 저장되어 있습니다. N개의 정수가 이미 큐에 들어가 있는 상태에서 평균을 구해야 합니다. 평균을 구하고 난 뒤, N개의 정수가 여전히 큐에 저장되어 있어야 합니다.  [Task] 큐에 1,2,3,4,5의 정수가 저장되어 있으며 저장된 순서는 중요하지 않습니다. 먼저, print 메소드(=큐에 저장된 정수를 순서대로 출력)를 구현하고, 호출하세요. 이 상태에서 avg() 메소드를 호출하고, 리턴값을 터미널에 출력하세요. 그리고 print 메소드를 다시 호출하세요.  터미널 출력 결과를 캡처하여 본 문서에 첨부하세요. 소스코드도 첨부파일로 제출해야 합니다. |

답변:

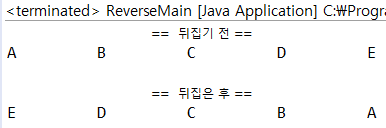
소스코드 : [Q3] 평균 구하기 소스코드.txt



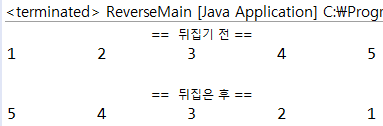
|  |
| --- |
| **[Q 4] 큐 뒤집기 (ft. Stack?) [배점: 20]**  ArrayQueue 또는 ListQueue를 사용해서, 큐에 저장된 Item을 뒤집는 public void reverse() 메소드를 구현하세요.  [Task 1] 큐에 E, D, C, B, A 문자가 순서대로 저장되어 있습니다. A가 가장 먼저 추가된 item이고, E가 가장 마지막에 추가된 item 입니다. print 메소드를 호출하세요 (= item을 순서대로 터미널에 출력). 이 상태에서 reverse() 메소드를 호출하세요. 다시 print 메소드를 출력하세요.  [Task 2] ‘Task 1’번을 반복하세요. 단, 이번에는 큐에 5, 4, 3, 2, 1 정수형 숫자가 저장되어 있습니다.  터미널 출력 결과를 캡처하여 본 문서에 첨부하세요. 소스코드도 첨부파일로 제출해야 합니다. |

소스코드 : [Q4] 큐 뒤집기 소스코드.txt

답변 (Task 1):



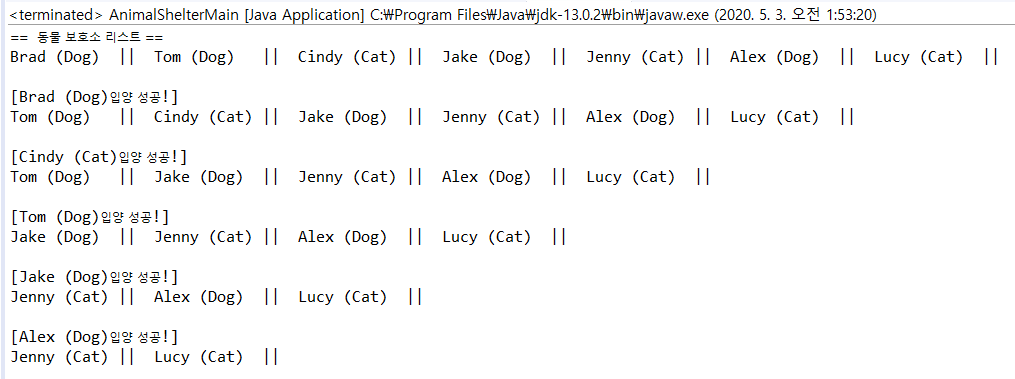
답변 (Task 2):



|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **[Q 5] 동물 보호소 [배점: 20]**  개와 고양이를 관리하는 동물 보호소 (AnimalShelter) 클래스를 구현하세요. 상세 내용은 강의노트를 참고하세요. print() 라는 메소드를 추가하세요. 이 메소드는 보호소에 있는 모든 동물을 순서대로 **한줄에** 출력합니다. 즉, 개와 고양이를 서로 다른 줄에 따로 출력하면 안되고, 한 줄에 모두 시간 순으로 출력해야 합니다. 왼쪽에는 가장 최근에 들어온 동물, 오른 쪽에는 가장 먼저 들어온 동물을 출력합니다. 각 동물을 출력할 때는 “이름(종류)” 의 형식으로 출력합니다. 예: Brad (Dog) – Tom (Dog) – (중간 생략) – Lucy (Cat) – Maggie (Cat)  다음과 같은 순서로 동물 보호소에 동물이 찾아왔습니다. 참고로, 표 상단에 있는 동물이 하단에 있는 동물보다 먼저 들어왔습니다.   |  |  |  | | --- | --- | --- | | **<이름>** | **<종류>** | **<참고>** | | Brad | 개 (Dog) | 가장 먼저 들어온 동물 | | Tom | 개 (Dog) |  | | Cindy | 고양이 (Cat) |  | | Jake | 개 (Dog) |  | | Jenny | 고양이 (Cat) |  | | Alex | 개 (Dog) |  | | Lucy | 고양이 (Cat) |  | | Maggie | 고양이 (Cat) | 가장 최근에 들어온 동물 |   [Task 1] 모든 동물을 추가한 후, print 메소드를 출력하세요.  [Task 2] adoptAny 호출 후 print  [Task 3] adoptCat 호출 후 print  [Task 4] adoptDog 호출 후 print  [Task 5] adoptAny 호출 후 print  [Task 6] adoptDog 호출 후 print  터미널/콘솔 출력 결과를 캡처하여 본 문서에 첨부하세요. 소스코드도 첨부파일로 제출해야 합니다. |

답변:

소스코드 : [Q5] 동물 보호소 소스코드.txt



**끝! 수고하셨습니다 ☺**