**Report03**



|  |  |
| --- | --- |
| **과 목 :** | **소프트웨어프로젝트** |
| **제출 일자 :** | **2019.05.13** |
| **담당 교수 :** | **이남규** |
| **학 과 :** | **사회복지학과** |
| **학 번 :** | **20170299** |
| **이 름 :** | **김유빈** |

1. 중심을 나타내는 정수 x, y와 반지름 radius 필드를 가지는 Circle 클래스를 작성하라. equals() 메소드는 두개의 Circle 객체의 중심이 같으면 같은 것으로 판별하도록 한다.

<핵심 문재>

main함수에서 Circle 클래스 작성의 힌트를 얻을 수 있다.

(1) Circle a = **new** Circle(2,3,5); // 중심 (2,3)에 반지름 5인 원

Circle b = **new** Circle(2,3,30); // 중심 (2,3)에 반지름 30인 원

-> 생성자 함수를 따로 작성해야 함을 알 수 있다.

(2) System.***out***.println("원 a : " + a); System.***out***.println("원 b : " + b);

-> 객체를 "Circle(2,3) 반지름 5"와 같은 형태로 출력시키는 toString() 메소드 작성해야 한다.

(3) **if**(a.equals(b))

System.***out***.println("같은 원");

**else**

System.***out***.println("서로 다른 원");

-> 두 개의 Circle 객체의 중심이 같으면 같은 것으로 판별하는 equals() 메소드 작성해야 한다.

**1) 생성자 함수 작성**

매개 변수 x, y, radius를 받아서 객체 변수 값을 초기화 시킨다.

**public** Circle(**int** x,**int** y,**int** radius) {

**this**.x = x;

**this**.y = y;

**this**.radius = radius;

}

this를 통해 이름이 같은 매개변수가 들어와도 객체 자신의 주소를 가리키도록 도왔다.

**2) equals() 메소드 작성**

매개변수는 Object obj 로 받는다. 이는 현재 객체와 비교할 객체를 받아들이기 위해서이다.

Object는 모든 클래스의 수퍼 클래스이므로 어떤 타입의 클래스든 인수로 받을 수 있다.

단, Circle 클래스로 타입 변환을 시켜서 사용해야 한다.

**public** **boolean** equals(Object obj) {

Circle p = (Circle)obj;

**if**(x == p.x && y == p.y) **return** **true**;

**else** **return** **false**;

}

if 를 통해 현 객체의 중심 값과 인수로 들어온 비교 객체의 중심 값이 같으면 true 반환, 그렇지 않으면 false 반환

**3) toString() 메소드 작성**

toString()은 기본적으로 “클래스 이름 + @ + 해시코드”이다. 이를 재작성해야 한다

Circle(2,3) 반지름 5 이와 같은 형식으로 출력되도록 작성한다.

**public** String toString() {

**return** "Circle(" + x + "," + y + ") 반지름 " + radius;

}

<결과 화면>

스크린샷이(가) 표시된 사진

자동 생성된 설명

1. MyString 클래스의 구현: String 클래스와 유사한 나만의 문자열 클래스로써 MyString 클래스를 만들고, MyStringEx 클래스를 이용하여 실행되도록 구현하시오.

<핵심 문제>

정해진 main함수에서 해결해야 할 문제들을 찾을 수 있다.

(1) MyString s = **new** MyString(**new** **char**[] {'a', 'B', 'c', 'D'});

-> 문자 배열을 입력받는 MyString 생성자 함수

(2) MyString.*println*(s);

-> MyString 객체를 바로 출력하는 문자열 출력 함수 println(MyString str)

(3) s.length()

-> 문자열의 길이 리턴하는 함수

(4) s.equals(s1)

-> 두 개의 문자열을 비교하는 함수 equals(Object obj), 같으면 true출력

(5) s.charAt(1)

->int index에 해당하는 문자 리턴하는 함수 charAt(int index)

(6) MyString s3 = s.toUppercase();

-> 문자열을 대문자로 변형하는 함수

(7) MyString s4 = s.substring(1, 2);

-> beginindex부터 시작하여 count만큼의 문자열만을 추출하는 함수

(8) MyString.*valueOf*(345)

->int형 값을 문자열로 변형하는 함수

(9) MyString.*valueOf*(**true**)

-> boolean형을 문자열로 변형하는 함수

**1)생성자 함수**

일단 MyString 클래스에서 char [] c;를 만든다.(문자 배열에 대한 레퍼런스 변수)

왜냐하면 문자열 == 문자의 배열

**public** MyString(**char** [] c) {**this**.c = c;}

매개로 받은 문자 배열을 해당 문자 배열 레퍼런스 변수가 가리키도록 한다.

**2)문자열 출력 함수**

문자들의 배열에서 문자 하나하나를 print()로 출력하고

마지막에 println()으로 엔터키 효과를 준다.

**public** **static** **void** println(MyString str){

**for**(**char** ch : str.c)

System.***out***.print(ch);

System.***out***.println();

}

for-each문을 사용하여 배열 요소 하나하나에 접근을 용이하게 한다!

**3)문자열 길이 함수**

for-each문을 사용하여 문자열 길이를 구한다

문자열 길이에 해당하는 변수 count 생성한다.

**public** **int** length() {

**int** count = 0;

**for**(**char** ch : c)

count++;

**return** count;

}

**4)문자열 비교 함수**

매개로 받는 MyString 객채와 현재 객체의 문자 배열을 하나하나 비교한다,

만약 하나라도 같지 않다면 바로 false 출력

for문을 다 돌았다면 true를 출력한다.

**public** **boolean** equals(Object obj) {

MyString p = (MyString)obj; **int** i = 0;

**for**(**char** ch : p.c)

**if**(**this**.c[i++] != ch) **return** **false**;

**return** **true**; }

**5) 문자 추출 함수**

매개로 입력받은 int index를 통해 문자들의 배열에서 해당 문자를 찾기가 수월하다.

**public** **char** charAt(**int** index) {**return** c[index];}

**6)대문자 변환 함수**

대문자 문자를 저장할 새로운 배열(upchr)을 생성한다.

만약에 문자가 소문자 이면 대문자로 변형하여 새로운 배열에 집어 넣는다.

단, 문자들 배열에서 문자 하나하나에 접근할 때 for-each문을 쓴다.

마지막에는 upchr에 대한 새로운 MyString 객체를 생성하여 주소를 반환한다.

**public** MyString toUppercase() {

**char** [] upchr = **new** **char** [**this**.length()];//현 객체와 똑같은 길이 == this.length()

**int** i = 0;

**for**(**char** ch : **this**.c)

**if**(ch >= 'a')//소문자라면

upchr[i++] = (**char**)(ch - 32); //대문자로 변형하여 문자 배열에 삽입 **else** //대문자라면

upchr[i++] = ch;

MyString s = **new** MyString(upchr);

**return** s;

}

**7)문자열 추출 함수**

새로운 MyString 객체를 생성하여 주소값을 돌려주므로 새로운 문자 배열 레퍼런스 변수(substr)가 필요하다.

시작 인덱스와 문자열 개수를 매개변수로 받아서 시작 인덱스에서 해당 개수만큼 문자열을 출력하는 함수이다.

시작 인덱스(int beginindex)만큼 건너뛰어서 문자에 접근하면 된다.

**public** MyString substring(**int** beginIndex, **int** count) {

**char** [] subchr = **new** **char** [count];

**for**(**int** i = 0; i<count ;i++)

subchr[i] = **this**.c[i+beginIndex]; //해당 객체 문자열의 beginIndex부분 부터 복사

MyString s = **new** MyString(subchr); **return** s; }

**8) int형을 문자열로 변형 함수 ★★★**

매개 변수로 들어오는 int형 값(int value)를 문자열로 변형하는 함수이다.

먼저 value의 자릿수를 구해야 한다(ex. 세자리)

변수count를 통해 value를 10씩 나누어 일의 자리가 나올 때까지 나눈다 -> 숫자 자릿수 구함

**int** count = 0;

**for**(**int** i = 1;;i\*=10) {

count++;

**if**(value/i >= 1 && value/i <=9)

**break**;

}

그 다음에는 숫자 자릿수 만큼 새로운 문자 배열 레퍼런스 생성한다(char [] val)

자릿수만큼 나눌 수 int ten 생성한다,

입력값 value = 345 일때,

(1) ten = 100

-> val[i] = (**char**)(value/ten + '0')-> 즉, 문자 ‘3’이 들어간다!

->value -= (value/ten)\*ten

value = 345 – 300

=45

(2)ten = 10

-> 문자 ‘4’가 문자열로 들어오고

-> value = 45 – 40

= 5

(3)ten = 1

-> 문자 ‘5’가 문자열에 들어오고 끝! ( 그 뒤로 value 값은 중요하지 않다)

마지막으로 char []val 에 대한 새로운 MyString 객체 생성하여 주소값을 반환한다.

**9) boolean형을 문자열로 변형 함수**

boolean형은 크게 true, false 두가지 경우밖에 없으므로

if문을 사용하여 바로 새로운 객체를 생성하면 된다.

**public** **static** MyString valueOf(**boolean** value) {

MyString s; //안그러면 오류가 뜬다!!!!

**if**(value)

s = **new** MyString(**new** **char**[] {'t', 'r', 'u', 'e'});

**else**

s = **new** MyString(**new** **char**[] {'f','a','l','s','e'});

**return** s;

}

<출력 화면>

스크린샷이(가) 표시된 사진

자동 생성된 설명

1. 도시 이름, 위도, 경도 정보를 가진 Location 클래스를 작성하고, 도시이름을 ‘키’로 하는 HashMap<String, Location> 컬렉션을 만드시오. 사용자로부터 입력 받아 4개의 도시를 아래 실행 결과를 참고하여 실행하라. 그리고 도시의 이름으로 검색하는 프로그램을 작성하라.

<핵심 문제>

(1) 도시 이름, 위도, 경도 정보를 가진 Location 클래스를 작성하기

(2) 도시이름을 ‘키’로 하는 HashMap<String, Location> 컬렉션을 만들기

(3)프로그램 작성: 사용자로부터 4개의 도시를 입력 받는다.

입력 받은 모든 도시들을 출력한다.

원하는 도시 이름을 입력하면 해당 도시 이름, 위도, 경도 정보 출력한다.

**1) Location 클래스**

Location 클래스에서 문자열 도시 이름(city), 위도(lat), 경도(longi) 변수와

이들을 매개 변수로 받는 생성자 함수 작성

**class** Location{

String city, lat, longi;//도시 이름, 위도와 경도

**public** Location(String city, String lat, String longi) {

**this**.city = city;

**this**.lat = lat;

**this**.longi = longi;

}

}

**2) HashMap<String, Location> 컬렉션**

main함수 안에서 HashMap<String, Location> 컬렉션 선언

HashMap<String, Location> map = **new** HashMap<String, Location>();

3) 프로그램 작성

① HashMap<String, Location> 컬렉션에 입력 받은 도시 이름과 위도, 경도를 삽입

“서울, 37, 126” 이와 같은 형식으로 입력을 받는 String s[](문자열 배열) -> 문자열 함수 중에서split()함수로 문자열을 “, “단위로 나누어 저장

s[0]: 도시이름, s[1]: 위도, s[2]: 경도에 해당함

이를 순서대로 컬렉션에 삽입

**for**(**int** i=0;i<4;i++) { //총 4번 입력 받는다

System.***out***.print(">> ");

Scanner scanner = **new** Scanner(System.***in***);

String str = scanner.nextLine();

String s[] = str.split(", "); //입력 받은 문자열을 ", "단위로 분리하기

map.put(s[0], **new** Location(s[0],s[1],s[2])); //도시,경도,위도를 컬렉션에 입력

}

② 컬렉션에 담은 내용들을 출력

HashMap 함수인 keySet()를 이용하여 해시맵의 모든 키를 담은 Set<K>컬렉션 생성한다.

이 Set<K>컬렉션을 다시 Iterator 컬렉션이 받아서 순차적으로 접근한다.

Set set = map.keySet();//해시맵의 모든 키를 담은 Set<K>컬렉션 생성하여 이를 받아주는 set 객체 생성

Iterator <String> it = set.iterator();//Iterator컬렉션으로 Set<K>컬렉션에 순차적으로 접근하기

키 컬렉션 값은 Iterator를 통해 가져와서 해시맵에서 해당 키에 대한 값들(도시 이름, 위도, 경도)를 가져온다.

**while**(it.hasNext()) {

String key = it.next();//Set<K>컬렉션 의 key값을 문자열 변수에 순차적으로 저장

Location locate = map.get(key); //해당 key에 대한 값(Location 객체)를 가져오기

System.***out***.println(locate.city + ", " + locate.lat + ", " + locate.longi);

}

③ 입력 받은 도시 이름(키)에 해당하는 정보 가져오기

사용자로부터 도시 이름을 입력 받는다(String key)

해시맵에서 key에 해당하는 값을 불러 온다. -> 해당 하는 key가 해시맵에 없으면 “null”생성

값이 있으면 이를 화면에 출력

Scanner s = **new** Scanner(System.***in***);

String key = s.nextLine();

Location locate = map.get(key); //키에 해당하는 값(Location 객체) 생성하기

**if**(locate == **null**) //해시맵에 입력 받은 키가 없다면

System.***out***.println(key + " 없습니다.");

**else**

System.***out***.println(locate.city + ", " + locate.lat + ", " + locate.longi);

사용자가 “종료”를 입력하면 프로그램을 종료함

<결과 화면>

텍스트이(가) 표시된 사진

자동 생성된 설명