1. Class 클래스

* 생성된 클래스와 인터페이스의 정보를 관리해주는 클래스
* getClass(), forName()메소드를 이용하여 Class 객체 얻음

Class cla = 참조변수.getClass();

Class cla2 = Class.forName(“패키지이름.클래스명");

Class cla3 = 클래스명.class;

1. String 클래스

* charAt(int index) : index번째의 문자를 char형으로 반환
* length() : 문자열의 길이를 int형으로 반환
* equals() : 문자열 비교할 때 사용하는 메소드
* indexOf(String str) : 해당문자열의 index를 반환

해당문자열을 찾지 못할경우 -1 반환

* replace(target, str) : target이 되는 문자열을 str로 치환
* substring(int beginindex) : beginindex부터 문자열 끝까지 자름

substring(int beginindex, int endindex) : beginindex부터 endindex까지 문자열 자름

* split(String regex) : 특정문자 기준으로 문자열을 나누고 나눈 문자열을

문자열 배열로 반환

* toLowerCase() : 알파벳을 소문자로 치환하여 반환
* toUpperCase() : 알파벳을 대문자로 치환하여 반환
* trim() : 문자열 앞뒤의 공백을 제거하고 공백이 제거된 문자열 반환
* valueOf() : 기본타입(int, short, double, char....)의 값을 문자열의 값으로 치환

1. Wrapper 클래스

* 자바에서 제공하는 기본타입들의 객체를 생성할 수 있는 클래스
* Integer, Character, Doublie, Boolean, Short......
* parse기본타입(String str) : 문자열을 기본타입의 변수로 치환

1. Math 클래스

* 수학 계산에 용이한 메소드들을 다수 제공하는 클래스
* abs(int(double) a) : 매개변수로 입력된 정수나 실수의 절대 값을 반환
* ceil(double a) : 매개변수로 입력된 실수의 올림값을 반환
* floor(double a) : 매개변수로 입력된 실수의 버림값을 반환
* round(double a) : 매개변수로 입력된 실수의 반올림값을 반환
* max(int(double) a, int(double) b) : 두 개의 정수나 실수 중 큰 값을 반환
* min(int(double) a, int(double) b) : 두 개의 정수나 실수 중 작은 값을 반환

1. java.util 패키지

* 개발할 때 자주 사용하는 자료구조(컬렉션)나 날짜 등 유용한 클래스 지원

1. Date 클래스

* 날짜와 시간정보를 저장하는 클래스
* SimpleDateFormat() : 날짜를 원하는 형식의 문자열로 변환할 수 있음

1. Calendar 클래스

* 운영체제 날짜와 시간을 얻을 때 사용하는 클래스
* 달력형태로 표시되기 때문에 요일정보도 얻을 수 있다.
* getInstance() : 현재 운영체제에 설정돼있는 시간대를 기준으로 객체 하나 생성
* get() : 생성된 Calendar객체에서 년, 월, 일, 요일 등 정보를 추출

1. 컬렉션 프레임워크

* 자료구조를 사용하여 객체들을 효율적으로 추가, 삭제, 검색할 수 있도록 구현해놓은

인터페이스와 클래스

* 컬렉션 프레임워크는 java.util 패키지에 구현되어 있음
* 인터페이스 : List, Set, Map
* List를 상속받은 구현 클래스 : ArrayList, Vector, LinkedList
* Set을 상속받은 구현 클래스 : HashSet, TreeSet
* Map을 상속받은 구현 클래스 : HashMap, HashTable, TreeMap, Properties

1. List 컬렉션

* List 저장되는 객체들을 인덱스로 관리(배열과 비슷)
* List는 배열과 다르게 가변성을 띄고 있음
  + 데이터 추가 시 길이 자동 증가
* add(E e) : List 마지막 요소 뒤에 e객체 추가
* add(int index, E e) : 지정된 index에 e객체 추가
* contains(Object o) : 매개변수로 입력된 객체가 저장되어 있는 지 조사
* get(int index) : 지정된 index에 저장되어 있는 객체 반환
* isEmpty() : List가 비어있는지 아닌지 조사(비어있으면 true, 비어있지 않으면 false 반환)
* size() : List에 저장된 객체개수를 반환
* clear() : List에 저장된 객체를 모두 삭제
* remove(int index) : 지정된 index에 저장된 객체 삭제
* remove(Object o) : List에 저장된 객체 중 매개변수로 입력된 객체를 삭제

1. ArrayList

* List<E> list = new ArrayList<E>();
  + E : List에 저장될 객체타입

Ex) List<String> list = new ArrayList<String>();

List<HashMap> list = new ArrayList<HashMap>();

1. Vector

* ArrayList와 구조는 동일
* Vector 한 작업에서 호출했을 때 그 작이 끝나기전까지는 다른 작업에서 호출 불가능

(스레드 동기화)

* ArrayList는 여러작업에서 호출가능 비동기화
* 비동기화된 ArrayList 속도가 빠름

1. LinkedList

* List에 저장되는 앞뒤 객체들과 연결구조를 생성하는 List
* 순차적으로 요소를 모두 검색하기 때문에 검색속도가 ArrayList보다 느림
* LinkedList는 객체를 중간에 추가/삭제하는 속도가 빠르다
* LinkedList는 중간에 객체 추가/삭제 시 앞뒤 연결관계만 바꿔주면됨
* ArrayList는 중간에 객체 추가/삭제 시 저장되어 있는 모든 객체의 인덱스를 재설정
* 순차적 객체 추가/삭제 : ArrayList > LinkedList
* 중간에 객체 추가/삭제 : ArrayList < LinkedList
* 검색 : ArrayList > LinkedList

1. Set 컬렉션

* List컬렉션은 저장순서를 유지하고 중복된 객체를 저장할 수 있지만

Set컬렉션은 저장순서가 없고 중복된 객체의 저장이 불가능

* add(E e) : 주어진 객체 저장
* constains(Object o) : 매개변수로 입력받은 객체가 포함되어 있는지 여부
* isEmpty() : 해당 Set컬렉션이 비어있는지 아닌지 판단(비어있으면 true, 비어있지 않으면 false)
* size() : 저장된 객체의 개수를 반환
* clear() : Set 컬렉션에 저장된 모든 객체 삭제
* remove(Object o) : 매개변수로 입력된 객체 삭제
* 저장순서가 유지되지 않기 때문에 index로 접근 불가능
* Iterator라는 객체를 사용하여 Set컬렉션에 데이터가 있는지 없는지 판단
  + Iterator에서 제공하는 메소드

hasNext() : 다음에 꺼내올 객체가 있는지 여부 판단(있으면 true, 없으면 false)

next() : Set 컬렉션에서 객체 하나를 꺼내옴

remove() : Set 컬렉션에서 객체 하나 제거

* + Iterator 객체 얻기

Set<E> set = new ….

Iterator<E> iterator = set.iterator();

if(iterator.hasNext()) {

if(iterator.next().equals(“홍길동”)) {

Str = iterator.next();

}

}

1. HashSet

* Set 인터페이스를 상속받아 구현한 가장 대표되는 클래스
* Set 컬렉션의 속성을 따라서 중복 값이 저장이 안되고 인덱스가 지정되지 않음

1. Map 컬렉션

* Map 컬렉션은 키값과 키값에 매핑되는 밸류값을 가지는 자료구조 형
* put(K key, V value) : 키값과 밸류값을 모두 입력하여 객체 추가
* containsKey(Object key) : 매개변수로 입력된 키 값이 있는지 여부(있으면 true, 없으면 false)
* containsValue(Object value) : 매개변수로 입력된 밸류 값이 있는지 여부(있으면 true, 없으면 false)
* get(Object key) : 매개변수로 입력된 키 값에 해당하는 밸류 값 반환
* size() : 저장된 키 값의 총 개수 반환
* isEmpty() : Map 컬렉션이 비어있는지 여부(비어 있으면 true, 비어 있지 않으면 false)
* clear() : 저장된 모든 키 값과 밸류 값 삭제
* remove(Object key) : 매개변수로 입력된 키 값과 키 값에 해당하는 밸류 값 삭제
* HashMap이 Map 컬렉션 중 가장 많이 사용됨