CSE2016 Programming Methodology

Week 8: Arrays

Instructor: Jinyoung Han (jinyounghan@hanyang.ac.kr)

HANYANG UNIVERSITY

Contents

HANYANG UNIVERSITY

Today's Schedule

1. Why We Need Arrays
2. Array Initialization
3. Bank Account Management
4. Database
5. Card Game
6. Two-Dimensional Arrays

01. Why We Need Arrays

HANYANG UNIVERSITY

A Problem

- A problem
  - Input: 6 scores from 6 students
  - Output: highest score?
- A simple way
  - int score1, score2, score3, score4, score5, score6;
  - int high_score = score1;
  - if(score2 > high_score) high_score = score2;
  - ...
  - if(score6 > high_score) high_score = score6;
- A better way?
  - for (i = 2; i <= 6; i++)

    { if(score$_i$ > high_score) high_score = score$_i$;}

01. Why We Need Arrays

HANYANG UNIVERSITY

Array

- Definition
  - A data structure that contains a fixed number of data with same type
- Syntax: using []
  - E.g., int[], String[]
- An array in Java is an object
  - Created by “new”
  - E.g., int[] r = new int[6];

r[0]	r[1]	r[2]	r[3]	r[4]	r[5]

01. Why We Need Arrays

HANYANG UNIVERSITY

Array Indexing

```
int[] r = new int[6];
int x = 6;

r[1] = 7;
r[3] = r[x-5] + 2;
int[] s = r;
```

0	7	0	9	0	0

02. Array Initialization

HANYANG UNIVERSITY

Initialization

- By default, elements in an array are initialized as:
  - int: 0, double: 0.0, Boolean: false, object type: null
- User initialization
  - int[] r = {1, 2, 4, 8, 16, 32};
- Using loops for initialization

```
int[] r3 = new int[12];
r3[0] = 1;
r3[1] = 1;
for (int i=2; i<r3.length; i=i+1)
{
        r3[i] = r3[i-1] + r3[i-2];
}
```

02. Array Initialization

HANYANG UNIVERSITY

Array as Arguments

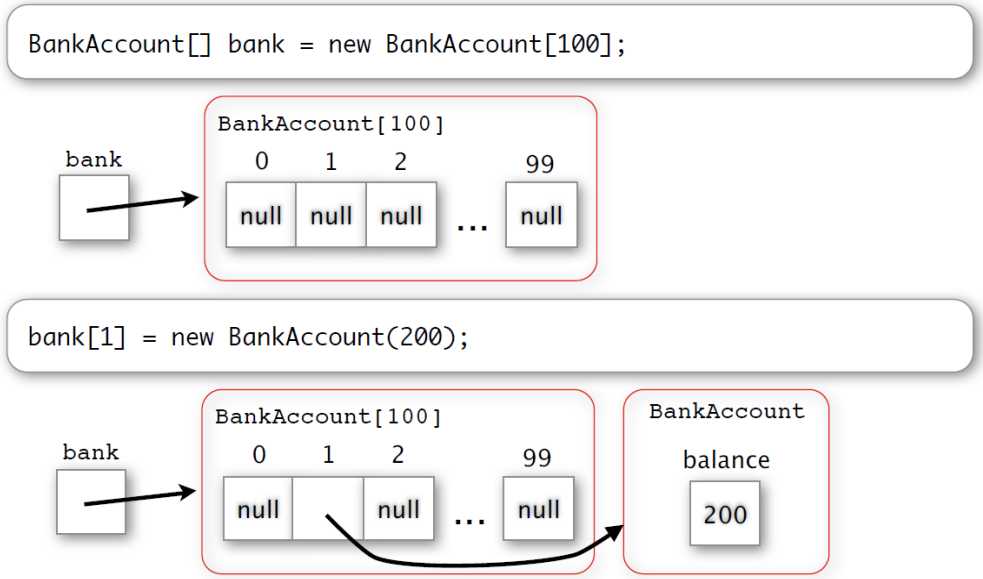
- Goal
  - Let's write a program that reverse the given array

```
static int[] reverse(int[] r)
{
    int size = r.length;
    int[] answer = new int[size];

    for(int i=0; i<size; i++)
    {
        answer[size-1-i] = r[i];
    }
    return answer;
}
```

HANYANG UNIVERSITY

An Example

- Can we manage 100 bank accounts?
  - We can use arrays!
- Creating an array for 100 accounts
  - BankAccount[] bank = new BankAccount[100];
- Creating a specific account
  - bank[1] = new BankAccount(200);
- Deposit
  - bank[1].deposit(600);
- Destroy
  - bank[1] = null;



HANYANG UNIVERSITY

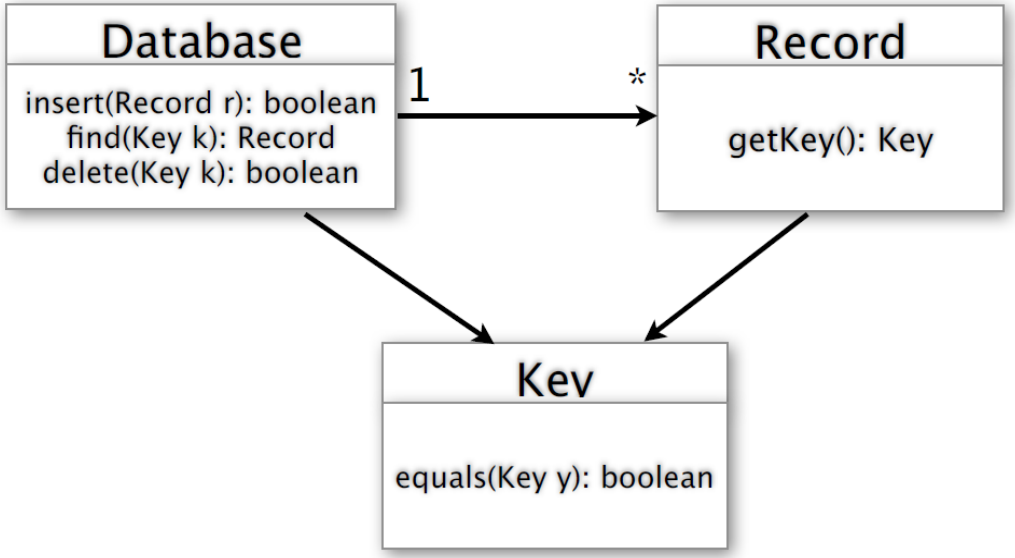
Database

- Database
  - A set of information
  - E.g., book information in a library, student data in a school
- Record
  - A unit for storing an information in database
  - Records are distinguished by “keys”

HANYANG UNIVERSITY

Simple Database

- A key is an object
- A record is an object
  - Containing a key object
- A database is an array for records
  - insert: inserting a record
  - find: finding a record
  - delete: deleting a record



HANYANG UNIVERSITY

Specification

class Database	
methods:	
insert(Record r): boolean	Inserting r
find(Key k): Record	Finding a record whose key is k
delete(Key k): boolean	Deleting a record whose key is k

class Record	
methods:	
keyOf(): Key	Returning the key of the record

class Key	
methods:	
Equals(Key m): boolean	Comparing m with itself

Constructor

```
public class Database
{
    private Record[] base;
    private int NOT_FOUND = -1;

    public Database (int initial_size)
    {
        if (initial_size <= 0)
                initial_size = 1;
        base = new Record[initial_size];
    }
```

Find-related

```
private int findLocation(Key k)
{
     for (int i=0; i<base.length; i++)
          if(base[i]!=null && base[i].keyOf().equals(k))
                    return i;
     return NOT_FOUND;
}

private int findEmpty()
{
     for (int i=0; i<base.length; i++)
          if(base[i]==null)
                    return i;
     return NOT_FOUND;
}

public Record find(Key k)
{
     int index = findLocation(k);
     if(index != NOT_FOUND)
          return base[index];
     else
          return null;
}
```

Delete

```
public boolean delete(Key k)
{
    int index = findLocation(k);
    if(index != NOT_FOUND)
    {
        base[index] = null;
        return true;
    }
    else
        return false;
}
```

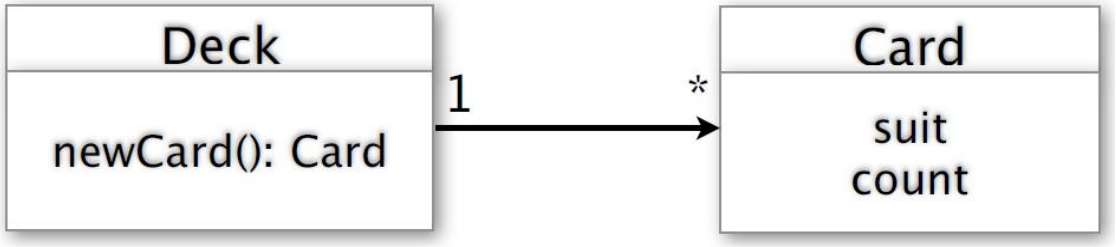
Insert

```
public boolean insert(Record r)
{
    if(findLocation(r.keyOf()) != NOT_FOUND)
        return false;
    int index = findEmpty();
    if(index != NOT_FOUND)
        base[index] = r;
    else
    {
        Record[] temp = new Record[base.length * 2];
        for(int i=0; i<base.length; i++)
                temp[i] = base[i];
        temp[base.length] = r;
        base = temp;
    }
    return true;
}
```

HANYANG UNIVERSITY

Card Game

- Goal
  - Return a card from a card deck
- Card
  - Suit: diamonds, hearts, clubs, spades
  - No: ACE(1), 2~10, JACK(11), QUEEN(12), KING(13)
- Card deck
  - An array



05. Card Game

HANYANG UNIVERSITY

Specification

class CardDeck	
attributes:	
private Card[] deck	Containing remaining cards
methods:	
newCard(): Card	Handing over a card
moreCards(): boolean	Return whether there are remaining cards

class Card	
attributes:	
private suit: String	Shape
private count: int	No.
methods:	
getSuit(): String	Returning shape
getCount(): int	Returning No.

05. Card Game

HANYANG UNIVERSITY

Card

```
public class Card {
	public static final String SPADES = "spades";
	public static final String HEARTS = "hearts";
	public static final String DIAMONDS = "diamonds";
	public static final String CLUBS = "clubs";
	public static final int ACE = 1;
	public static final int JACK = 11;
	public static final int QUEEN = 12;
	public static final int KING = 13;
	public static final int SIZE_OF_ONE_SUIT = 13;

	private String suit;
	private int count;
	public Card(String s, int c)
	{
		suit = s;
		count = c;
	}

	public String getSuit()
	{
		return suit;
	}

	public int getCount()
	{
		return count;
	}
}
```

CardDeck

```
public class CardDeck
{
    private int card_count;
    private Card[] deck = new Card[4*Card.SIZE_OF_ONE_SUIT];

    private void createSuit(String which_suit)
    {
        for(int i=1; i<=Card.SIZE_OF_ONE_SUIT; i++)
        {
            deck[card_count] = new Card(which_suit, i);
            card_count++;
        }
    }
    public CardDeck()
    {
        createSuit(Card.SPADES); createSuit(Card.HEARTS);
        createSuit(Card.CLUBS); createSuit(Card.DIAMONDS);
    }
```

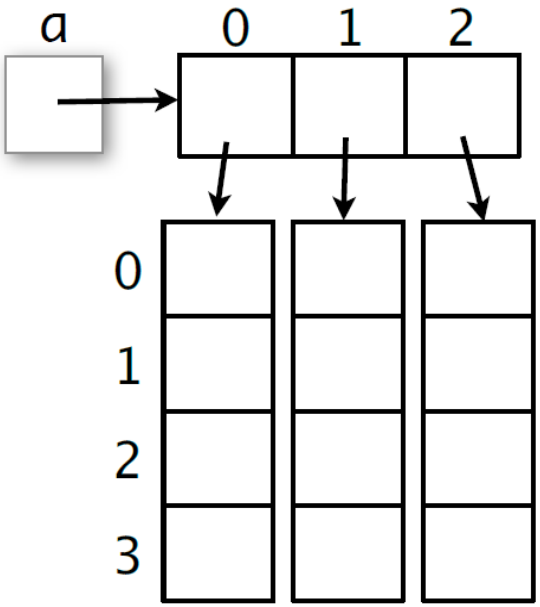
CardDeck

```
public Card newCard()
{
    Card next_card = null;
    if(card_count != 0 )
    {
        int index = (int)(Math.random() * card_count);
        next_card = deck[index];

        for(int i=index+1; i<card_count; i++)
                deck[i-1] = deck[i];
        card_count--;
    }
    return next_card;
}

public boolean moreCards()
{
    return card_count > 0;
}
```

HANYANG UNIVERSITY

Two-Dimensional

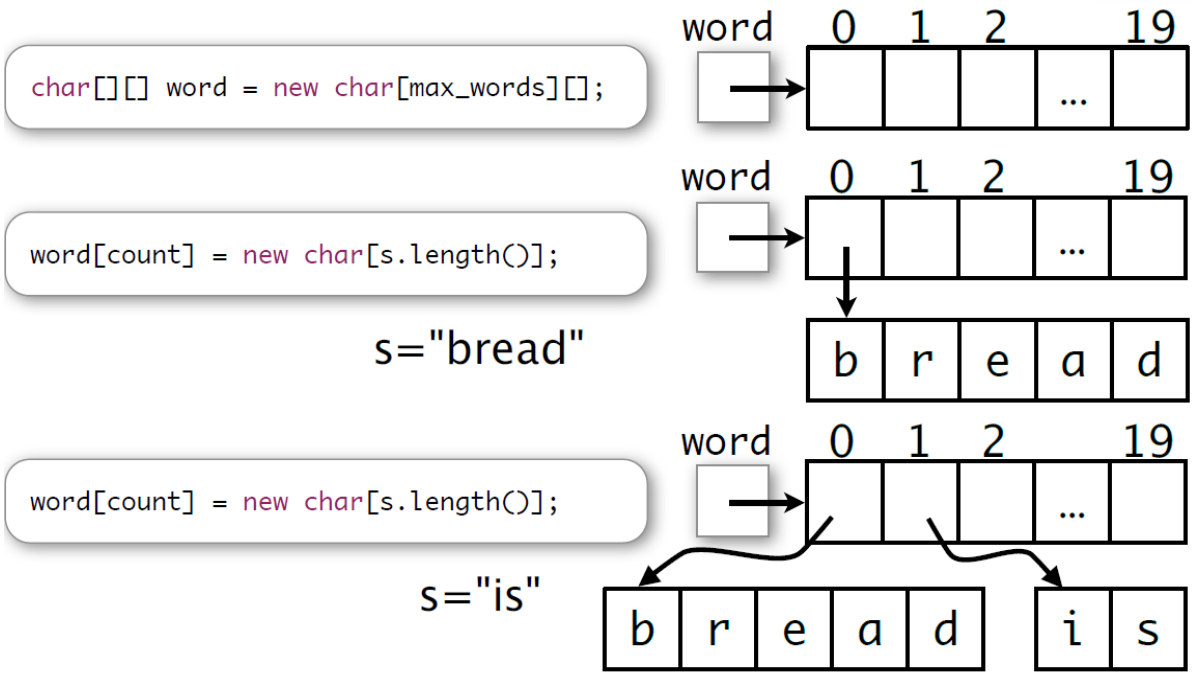
- Syntax
  - int[][] a = new int[3][4];
  - a.length -> 3
  - a[0].length -> 4



HANYANG UNIVERSITY

Ragged Array

- Each element in an array may have different size
- Example
  - int max_worlds = 20;
  - char[][] word = new char[max_words][];
  - word[0] = new char[s.length()];



HANYANG UNIVERSITY

Summary

1. Why We Need Arrays
2. Array Initialization
3. Bank Account Management
4. Database
5. Card Game
6. Two-Dimensional Arrays

Thanks

Week 8: Arrays

Instructor: Jinyoung Han (jinyounghan@hanyang.ac.kr)