Exam

Explanations

Carefully formulate your answers, code, and images

Formulate your answers precise and to the point.

Code shall be written so that it is easy to follow and understand. In some situations suitable comments can contribute to understanding. Small syntactical errors can be tolerated. If some parts of code cannot be exactly produced, it is possible that well-formed pseudocode may provide a solution. Do not write more code than necessary; if just a method is requested there is no need to create a whole class. All program code is to be written in Java.

When an array (vector) or an object is drawn, it must be clearly visible by which reference the array or object is referred to, and what data is located inside it. When an array or object contains a reference, the resource that is referred to (an object or array) shall be drawn. All references shall have relevant labels.

Points and grading

In total: 42 points

For grade E at least: 21 points

For grade D at least: 25 points

For grade C at least: 29 points

For grade B at least: 33 points

For grade A at least: 37 points

.

Tasks

Task 1 (2 points + 4 points)

public static void main (String[] args)

{

int[][] b = { {1}, {2, 3}, {4, 5, 6} };

int[] u = new int[b.length];

for (int i = 0; i < u.length; i++)

u[i] = b[i][b[i].length - 1];

int[][] v = transform (b);

}

public static int[][] transform (int[][] a)

{

int[][] w = new int[a.length][];

for (int i = 0; i < w.length; i++)

w[i] = a[a.length - 1 - i];

return w;

}

a) Draw the array referred to by the reference u.

b) Draw the array referred to by the reference v.

Task 2 (3 points + 3 points + 3 points)

The class Rectangle represents a rectangle:

class Rectangle

{

// the lengths of the rectangle’s sides

private double length;

private double width;

// the rectangle’s color

private String color;

public Rectangle (double length, double width, String color)

{

this.length = length;

this.width = width;

this.color = color;

}

public String getColor ()

{

return color;

}

public double area ()

{

return length \* width;

}

}

a) A static method, minRectangle, accepts an array of rectangles (objects of type Rectangle) and returns the rectangle with the smallest area. Create this method.

b) A static method, selectRectangles, accepts an array of rectangles (objects of type Rectangle) and a color (an object of type String), and returns those rectangles (as an array) that are of the given color. Create this method.

c) Create an array of rectangles (objects of type Rectangle). Then use the methods selectRectangles and minRectangle to determine the smallest of the rectangles in the array that are yellow.

Task 3 (3 points + 3 points + 3 points)

The class CharSequence manages a sequence of characters:

class CharSequence

{

private static final int INITIAL\_CAPACITY = 3;

// characters in the character sequence

private char[] chars;

// the number of characters

private int charCount;

public CharSequence ()

{

this.chars = new char[INITIAL\_CAPACITY];

this.charCount = 0;

}

public String toString ()

{

return "[" + chars.length + ", " + new String (chars, 0, charCount) + "]";

}

// add appends a given character to the character sequence.

// code is missing here

// replace replaces the character at a given index

// with a given character.

// If an invalid index is given, an exception of type

// java.lang.IndexOutOfBoundsException is thrown.

// code is missing here

}

An instance of class CharSequence is created and used like this:

CharSequence cs = new CharSequence ();

for (char c = 'a'; c <= 'e'; c++)

{

cs.add (c);

System.out.println (cs);

}

cs.replace (1, 'B');

System.out.println (cs);

cs.replace (5, 'B');

When this code segment is executed, the following printout is generated:

[3, a]

[3, ab]

[3, abc]

[6, abcd]

[6, abcde]

[6, aBcde]

Exception in thread "main" java.lang.IndexOutOfBoundsException: bad index

at CharSequence.replace(UseCharSequence.java:49)

at UseCharSequence.main(UseCharSequence.java:69)

a) Implement the method add.

b) Implement the method replace.

c) When the code segment has been executed, what does the object referred to by the reference cs look like? Draw this object.

Task 4 (4 points + 5 points)

The class MessageQueue defines a queue for messages. The classes LifoMessageQueue and FifoMessageQueue represent two different types of queues.

abstract class MessageQueue

{

protected String[] messages;

public MessageQueue ()

{

messages = new String[5];

}

// put inserts a given message in the queue.

public abstract void put (String message);

// take removes the next message from the queue and returns it.

public abstract String take ();

public String queueStructure ()

{

String s = "";

for (int index = 0; index < messages.length; index++)

if (messages[index] == null)

s = s + "-";

else

s = s + messages[index];

return s;

}

}

class LifoMessageQueue extends MessageQueue

{

private int lastIndex = -1;

public LifoMessageQueue ()

{

super ();

}

public String toString ()

{

String s = "";

for (int index = 0; index <= lastIndex; index++)

s = s + messages[index];

return s;

}

public void put (String message)

{

lastIndex = lastIndex + 1;

messages[lastIndex] = message;

}

public String take ()

{

String lastMessage = messages[lastIndex];

messages[lastIndex] = null;

lastIndex--;

return lastMessage;

}

}

class FifoMessageQueue extends MessageQueue

{

private int numberOfMessages = 0;

private int firstIndex = 0;

private int lastIndex = -1;

public FifoMessageQueue ()

{

super ();

}

public String toString ()

{

String s = "";

int index = firstIndex;

for (int i = 1; i <= numberOfMessages; i++)

{

s = s + messages[index];

index = (index + 1) % messages.length;

}

return s;

}

public void put (String message)

{

lastIndex = (lastIndex + 1) % messages.length;

messages[lastIndex] = message;

numberOfMessages++;

}

public String take ()

{

String firstMessage = messages[firstIndex];

messages[firstIndex] = null;

firstIndex = (firstIndex + 1) % messages.length;

numberOfMessages--;

return firstMessage;

}

}

a)

MessageQueue mq = new LifoMessageQueue ();

mq.put ("A");

mq.put ("B");

mq.put ("C");

mq.put ("D");

String m = mq.take ();

m = mq.take ();

mq.put ("E");

System.out.println (mq);

System.out.println (mq.queueStructure ());

Which printout is generated when this code segment is executed?

b)

MessageQueue mq = new FifoMessageQueue ();

mq.put ("a");

mq.put ("b");

mq.put ("c");

String m = mq.take ();

mq.put ("d");

mq.put ("e");

mq.put ("f");

m = mq.take ();

m = mq.take ();

System.out.println (mq);

System.out.println (mq.queueStructure ());

Which printout is generated when this code segment is executed?

Task 5 (5 points + 4 points)

An algorithm, that sorts a sequence of elements, can be illustrated as below:

[H, B, G, A, E, F, C, D]

[B, H, G, A, E, F, C, D]

[B, G, H, A, E, F, C, D]

[A, B, G, H, E, F, C, D]

[A, B, E, G, H, F, C, D]

[A, B, E, F, G, H, C, D]

[A, B, C, E, F, G, H, D]

[A, B, C, D, E, F, G, H]

[A, B, C, D, E, F, G, H]

a) Create a method sort that accepts an array of character strings, and sorts them according to the given algorithm.

b) Let *n* denote the number of elements to be sorted. Determine in such a case the time complexity for the algorithm in terms of the number of comparisons – both the best case and the worst case. Categorize the corresponding complexity functions: to which *Θ* –set do they belong?