Colin Kincaid Practice Midterm Solutions

CS 106A July 20, 2018

Solutions to Practice Midterm

Problem 1: Karel the Robot

**public class FarmerKarel extends SuperKarel {**

**public void run() {**

**checkRow();**

**while (leftIsClear()) {**

**moveToNextRow();**

**checkRow();**

**}**

**}**

**/\* Precondition: Karel is facing East with at least 1 row above it.**

**\* Postcondition: Karel is facing East one row up.**

**\*/**

**private void moveToNextRow() {**

**turnLeft();**

**move();**

**turnRight();**

**}**

**/\* Precondition: Karel is facing East at the beginning of a row.**

**\* Postcondition: Karel is in the same position, but has put all**

**\* of the beepers in the row on that square.**

**\*/**

**private void checkRow() {**

**if (frontIsClear()) {**

**goToBeeper();**

**while (beepersPresent()) {**

**bringBeeperHome();**

**goToBeeper();**

**}**

**turnAround();**

**moveToWall();**

**turnAround();**

**}**

**}**

**/\* Precondition: Karel is not facing a wall.**

**\* Postcondition: Karel has moved until it reaches a beeper or a wall.**

**\*/**

**private void goToBeeper() {**

**move();**

**while (frontIsClear() && noBeepersPresent()) {**

**move();**

**}**

**}**

**/\* Precondition: Karel is standing on a beeper, facing East**

**\* Postcondition: Karel is standing on the leftmost square of**

**\* the same row, facing East, with the beeper now on that square.**

**\*/**

**private void bringBeeperHome() {**

**pickBeeper();**

**turnAround();**

**moveToWall();**

**turnAround();**

**putBeeper();**

**}**

**/\* Precondition: NA**

**\* Postcondition: Karel has moved straight until it reaches a wall.**

**\*/**

**private void moveToWall() {**

**while (frontIsClear()) {**

**move();**

**}**

**}**

**}**

**Problem 2: Java Statements and Expressions**

**(2a)**

**1 + (2 + “B”) + ‘A’ \_\_\_\_\_\_”12BA”\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**11 / 2 > 5 || 5 % 2 == 1 \_\_\_\_\_\_\_\_true\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**(char)(‘B’ + 2) + “” + 4 + 27 / 3 \_\_\_\_\_\_\_”D49”\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**21 / 2.0 + 3 % 4 – 23 / 2 \_\_\_\_\_\_\_\_\_2.5\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**!(3 / 2 < 1.5) && (4 > 5 || 2 % 3 == 0) \_\_\_\_\_\_\_false\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**(2b)** What are the color, dimensions and location of **rect** on the canvas?

**x = 9**

**y = 9**

**width = 12**

**height = 35**

**color is red**

Problem 3: Console Programs

Note that exact output matching was not required as long as the functionality was correct.

public class MovieKiosk extends ConsoleProgram {

public void run() {

String movieNames = "";

double total = 0;

int voucher = 0;

String movieName = readLine("Movie name: ");

while (movieName.length() > 0) {

int numTickets = readInt("# tickets: ");

double ticketPrice = readDouble("Ticket price: ");

/\* If the voucher doesn't cover the total cost, add

\* the remaining balance to total. Otherwise,

\* the user owes nothing (the voucher covers it all)

\*/

if (voucher < ticketPrice \* numTickets) {

total += numTickets \* ticketPrice - voucher;

}

voucher = 0;

// Randomly award a voucher for the next purchase

if (RandomGenerator.getInstance().nextBoolean(0.1)) {

voucher = RandomGenerator.getInstance()

.nextInt(5, 25);

println("You've won a $" + voucher +

" voucher for your next purchase!");

}

// Add the movie name to our string

if (movieNames.equals("")) {

movieNames = movieName;

} else {

movieNames += " and " + movieName;

}

println();

movieName = readLine("Movie name: ");

}

println();

if (!movieNames.equals("")) {

println("Movies: " + movieNames);

println("Total: $" + total);

} else {

println("Movies: None");

}

}

}

Problem 4: Graphics Programs

public class StickHero extends GraphicsProgram {

private boolean isOriginalSize;

private GImage player;

public void run() {

isOriginalSize = true;

player = new GImage("res/player.png");

add(player, 0, getHeight() / 2.0 - player.getHeight() / 2.0);

// Animate the player across the screen

while (true) {

player.move(5, 0);

// If the player reaches the right edge, move to the left edge

if (player.getX() + player.getWidth() >= getWidth()) {

player.setLocation(0, getHeight() / 2.0 -

player.getHeight() / 2.0);

}

pause(30);

}

}

public void mouseClicked(MouseEvent e) {

GObject obj = getElementAt(e.getX(), e.getY());

if (obj == player) {

if (isOriginalSize) {

// double the size while keeping the center the same

player.setX(player.getX() - player.getWidth() / 2.0);

player.setY(player.getY() - player.getHeight() / 2.0);

player.setSize(2\*player.getWidth(), 2\*player.getHeight());

} else {

// half the size while keeping the center the same

player.setX(player.getX() + player.getWidth() / 4.0);

player.setY(player.getY() + player.getHeight() / 4.0);

player.setSize(0.5\*player.getWidth(),

0.5\*player.getHeight());

}

isOriginalSize = !isOriginalSize; // flip the boolean

}

}

}

Problem 5: Text Processing

(5a)

private String replaceMention(String str) {

if (str.length() == 0 || str.charAt(0) != '@') return str;

// If only one name, just remove the '@'

if (countUppercaseLetters(str) == 1) {

return str.substring(1);

}

// Build up a new string with the mention expanded

String newStr = "";

for (int i = 1; i < str.length(); i++) {

char ch = str.charAt(i);

// If it's upper case, check if it's the last uppercase letter

if (Character.isUpperCase(ch)) {

// If it's the last name, print out initial + '.' and return

if (countUppercaseLetters(str.substring(i+1)) == 0) {

newStr += " " + ch + ".";

return newStr;

} else {

/\* Otherwise, it's the start of a new middle name,

\* so add a space before.

\*/

newStr += " " + ch;

}

} else {

// Otherwise, append the character as normal

newStr += ch;

}

}

return newStr;

}

// A helper method that returns the number of uppercase letters in the given string.

private int countUppercaseLetters(String str) {

int count = 0;

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

if (Character.isUpperCase(str.charAt(i))) {

count++;

}

}

return count;

}

(5b)

public class ReplaceMentions extends ConsoleProgram {

public void run() {

String filename = promptUserForFile("Enter filename: ", "res");

try {

Scanner s = new Scanner(new File(filename));

// We can assume only 1 line, so we need only 1 scanner

while (s.hasNext()) {

print(replaceMention(s.next()) + " ");

}

s.close();

} catch (IOException e) {

println("File could not be read.");

}

}

}