Name/Surname: Id:

# CENG 112 – DATA STRUCTURES

## SPRING 2015-2016 / Midterm Exam I Solutions

01.04.2016

**Q1. (25 Points) Functions and Program Control**

**a)** Write a Java program that prints out the conversion table of temperatures from Fahrenheit to Celsius from 0°F to 150°F in increments of 5°F. (C = (F-32)/1.8).

**Answer:**

```
public class Fahr2Celcius {

    public static void main(String[] args) {
        for (int f = 0; f <= 150; f += 5) {
            double c = (f - 32.0) / 1.8;
            StdOut.printf("%6d.00  %6.2f\n", f, c);
        }
    }


}
```

**b)** What is the output of the following code fragment?

```
int [] arr = { 1, 2, 3, 4, 5 };
int [] brr = arr;
brr[3] += arr[2];
arr[2] += brr[3];

for (int i = 1; i < arr.length; i += 1)
      StdOut.println(brr[i-1]);

String s = arr[0] + “ wi” + arr[0] + arr[0]
s = s + “ come to the “ + arr[0] + “ectures on Friday.”;
StdOut.println(s);

for (int c = 0; c < 100; c += 1)
      if (c % 2 == 0 && c / 25 >= 2)
            StdOut.println(c);
```

**Answer:**

```
1
2
10
7
1 wi11 come to the 1ectures on Friday.
50
52
54
56
58
60
62
64
66
68
70
72
74
76
78
80
82
84
86
88
90
92
94
96
98
```

## Q2. (25 Points) Abstract Data Types

**a)** Write the API for a 2D Point ADT that will support the following operations:

- Create a point at the Cartesian coordinates (x, y)
- Create a point at the polar coordinates (r, theta)
- Translate the point dx units in the x-axis and dy units in the y-axis
- Compute the distance to another given Point object
- Convert the point to a String

**Answer:**

```
public class Point2D
                Point2D(double x, double y) // you could use long, int or float
static Point2D createFromPolar(double r, double theta)
void           translate(double dx, double dy)
double         distanceTo(Point2D p)
String         toString()
```

**b)** Given the following API for a graphics Pen, write a program that draws a rectangle of width 100, height 200 and top-left corner at (120, 150) and a circle at the center of the rectangle that will touch the top and bottom lines of the rectangle.

```
public class Pen
      Pen()                         // create a new Pen object
void  moveTo(double x, double y)    // move the pen to coordinates (x, y) without drawing
void  drawTo(double x, double y)    // draw a line from current pen position to (x, y) and
                                    // also move the pen to (x, y)
void  circle(double cx, double cy,
             double r)              // draw a circle of radius r centered at (cx, cy)
```

Name/Surname: Id:

**Answer:**

```
public class Draw {
      public static void main(String[] args) {
            Pen p = new Pen();
            p.moveTo(120, 150);
            p.drawTo(220, 150);
            p.drawTo(220, 350);
            p.drawTo(120, 350);
            p.drawTo(120, 150);
            p.circle(170, 250, 100);
      }
}
```

**Q3. (25 Points) Stacks**

**a)** What is the output of the following code fragment?

```
Stack<Integer> s0 = new Stack<Integer>();
Stack<Integer> s1 = new Stack<Integer>();

for (int j = 1; j <= 20; j += 1) {
      s0.push(j);
      s1.push(21-j);
}

while (s0.pop() > s1.pop())
      StdOut.print(s1.pop() + “:”);
```

**Answer:**

```
2:4:6:8:10:12:14:
```

**b)** Assume that we use a resizable array to implement a Stack ADT, the initial capacity is 4 and we push 100 elements to the stack one by one. How many times do we need to resize the array if we multiply the capacity by 2 whenever we have to grow the array? How many if the capacity is multiplied by 1.5 instead?

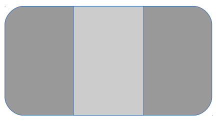
**Answer:**

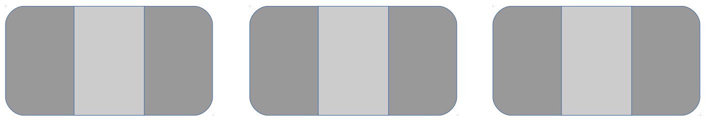
**Capacity Evolution for multiplier = 2, * denotes call to resize():**
**4 → 8* → 16* → 32* → 64* → 128***
**resize() will be called 5 times.**

**Capacity Evolution for multiplier = 1.5, * denotes call to resize():**
**4 → 6* → 9* → 13* → 19* → 28* → 42* → 63* → 94* → 141***
**resize() will be called 9 times.**

## Q4. (25 Points) Circular Doubly Linked List Implementation of Queues

A doubly linked list contains nodes that point both to the next and the previous elements in the list. In a circular list, the previous link of the first element points to the last element and the next link of the last element points to the first element. The following are two circular double linked lists containing a single element and three elements.





Given the following partial Queue implementation fill in the enqueue() method.

```
public class Queue<Item> {
       private Node head;
       private int size;
       private class Node { Item data; Node next; Node prev; }

       public Queue() {
              head = null;
              size = 0;
       }

       public int size() { return size; }
       public boolean isEmpty() { return size == 0; }
       public void enqueue(Item item) {

              Node n = new Node();
              n.data = item;

              if (size == 0) {
                     n.prev = n;
                     n.next = n;
                     head = n;
              } else {
                     n.next = head;
                     n.prev = head.prev;
                     head.prev.next = n;
                     head.prev = n;
                     head = n.next;
              }

              size += 1;

       }

       public Item dequeu() {
              Item r = head.data;
              size -= 1;

              if (size == 0) {
                     head = null;
              } else {
                     head.prev.next = head.next;
                     head.next.prev = head.prev;
                     head = head.next;
              }

              return r;
       }
}
```