

CSE 331 18wi Midterm Exam 2/8/18

Name _____ UW ID # _____

There are 11 questions worth a total of 120 points. Please budget your time so you get to all of the questions. Keep your answers brief and to the point.

The exam is closed book, closed notes, closed electronics, closed mouth, open mind.

Please be sure to write your answers in the spaces provided. Do not write on the backs of pages or on pages that indicate that no answers should be written there. These will not be scanned for grading. An extra blank page is provided at the end of the exam if you need additional space for your answers.

Many of the questions have short solutions, even if the question is somewhat long. Don't be alarmed.

For all of the questions involving proofs, assertions, invariants, and so forth, you should assume that all numeric quantities are unbounded integers (i.e., overflow cannot happen) and that integer division is truncating division as in Java, i.e., $5/3$ evaluates to 1.

If you don't remember the exact syntax of some command or the format of a command's output, make the best attempt you can. We will make allowances when grading.

Relax, you are here to learn.

Please wait to turn the page until everyone is told to begin.

Score _____ / 120

1. _____ / 12

7. _____ / 18

2. _____ / 12

8. _____ / 9

3. _____ / 15

9. _____ / 12

4. _____ / 12

10. _____ / 12

5. _____ / 4

11. _____ / 2

6. _____ / 12

CSE 331 18wi Midterm Exam 2/8/18

Remember: For all of the questions involving proofs, assertions, invariants, and so forth, you should assume that all numeric quantities are unbounded integers (i.e., overflow can not happen) and that integer division is truncating division as in Java, i.e., $5/3 \Rightarrow 1$.

Question 1. (12 points) (Forward reasoning) The traditional warmup question. Using forward reasoning, write an assertion in each blank space indicating what is known about the program state at that point, given the precondition and the previously executed statements. Your final answers should be simplified. Be as specific as possible, but be sure to retain all relevant information.

(a) (5 points)

```
{ y > 0 }  
  
x = 3;  
  
{ _____ }  
  
y = x + y;  
  
{ _____ }  
  
z = x * y;  
  
{ _____ }
```

(b) (7 points)

```
{ |x| < 3 }  
  
if (x < 0)  
  
    { _____ }  
  
    y = 3 + x;  
  
    { _____ }  
  
else  
  
    { _____ }  
  
    y = 2 * x;  
  
    { _____ }  
  
{ _____ }
```

CSE 331 18wi Midterm Exam 2/8/18

Question 2. (12 points) (Backward reasoning) The other traditional warmup question. Using backwards reasoning, find the weakest precondition for each sequence of statements and postcondition below. Insert appropriate assertions in each blank line. You should simplify your final answers if possible.

(a) (5 points)

```
{ _____ }  
b = a - 6;  
  
{ _____ }  
c = 2*a + b;  
  
{ c > 0 && b > 0 }
```

(b) (7 points)

```
{ _____ }  
if (x <= 2) {  
    { _____ }  
    y = x + 1;  
    { _____ }  
} else {  
    { _____ }  
    y = x - 2;  
    { _____ }  
}  
  
{ y > 0 && y < 3 }
```

CSE 331 18wi Midterm Exam 2/8/18

Question 3. (15 points) Proofs. The following method is supposed to return true if its integer argument n is a power of 2 (i.e., 2^k for some k) and false otherwise. Write a suitable invariant and appropriate assertions to prove that it works correctly.

Hint: an extra variable i is included in the code. It is not part of the algorithm, but it might be helpful in your proof. If it is not useful, you can ignore it.

```
/** @param n Input number.
 *   @requires n >= 1
 *   @return true if n = 2^k for some integer k. */
public boolean isPowerOfTwo(int n) {

    { pre: _____ }
    int i = 0; // extra variable that might be useful

    { inv: _____ }
    while(n%2 == 0) {

        { _____ }
        i = i + 1; // extra variable

        { _____ }
        n = n / 2;

        { _____ }
    } // end of loop

    { _____ }
    boolean answer = (n==1);

    { post: _____ }
    return answer;
}
```

CSE 331 18wi Midterm Exam 2/8/18

Question 4. (12 points, 3 points each) Specifications and implementations. Each of the following code fragments contains a short method specification and implementation. Each one contains an error: either (i) the specification itself is flawed because it is inconsistent, incomplete, or includes inappropriate things, or is contradictory [specerror]; or (ii) the specification is fine, but the implementation does not satisfy the specification [incorrectimpl]; or (iii) the code is implemented badly or is of poor quality or style or has other problems even though it satisfies the spec. [quality].

For each part, circle the error type, and give a brief (20 words or less) description of the problem and how to fix it. If there is more than one kind of error, identify and describe only the most important one. Specifications have been shortened to omit some parts and save space; the errors are not due to omitted JavaDoc parts in the specifications.

(a)

```
private Random random;

/** @return a random color.
 * @requires this.random is non-null
 */
public String getRandomColor() {
    String[] colors = { "blue", "red", "yellow" };
    return colors[this.random.nextInt(3)];
}
```

Error kind: specerror incorrectimpl quality

Reason and how to fix?

(b)

```
/** @param list a List of integers
 * @requires list has at least one element
 * @return the smallest integer in the list
 */
public int min(List<Integer> list) {
    Collections.sort(list);
    return list.get(0);
}
```

Error kind: specerror incorrectimpl quality

Reason and how to fix?

(continued on next page)

CSE 331 18wi Midterm Exam 2/8/18

Question 4. (cont.)

```
(c) public class IntQueue {
    private List<Integer> queue;

    /** @return the elements of this queue as a List.
     */
    public List<Integer> getIntegers() {
        return queue;
    }
    ...
}
```

Error kind: specerror incorrectimpl quality

Reason and how to fix?

```
(d) public class IntQueue {
    private List<Integer> queue;

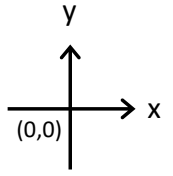
    /** @return number of elements in the queue.
     */
    public int length() {
        int length = 0;
        for (Integer i : queue) {
            ++length;
        }
        return length;
    }
    ...
}
```

Error kind: specerror incorrectimpl quality

Reason and how to fix?

CSE 331 18wi Midterm Exam 2/8/18

The next several questions concern two classes for representing points and rectangles on a 2-D plane. These classes use standard Cartesian x, y coordinates, with the origin in the center, x coordinates increasing to the right, and y coordinates increasing going up.



We are given a class `Point` with the following specifications (condensed to save space).

```
/** A Point is an immutable object that represents a
 * location (x,y) on a 2-D plane. */
public class Point {
    // rep and other private details omitted

    // Creators:
    /** Construct a new Point with coordinates (x,y) */
    public Point(int x, int y) { ... }

    // Observers: return x or y coordinate
    public int getX() { ... }
    public int getY() { ... }

    // equals/hashcode
    /** Return true if other is a Point with the same (x,y)
     * coordinates as this, otherwise return false */
    public boolean equals(Object other) { ... }

    /** Return an appropriate hash code for this */
    public int hashCode() { ... }
} // end of class Point
```

We will use this class `Point` to implement class `Rectangle`, which has the following overview:

```
/** A Rectangle is a mutable object on a 2-D plane. Its
 * corners are specified by two points on the plane ul and
 * lr, which give the coordinates of the upper-left and
 * lower-right corners of the Rectangle. All Rectangle
 * objects have a height and width of at least 1, i.e.,
 * a rectangle is not empty. That implies the coordinates of
 * the top and bottom differ by at least one, and also the
 * coordinates of the left and right sides differ by at
 * least one. It is possible to change the coordinates of
 * the corners of a Rectangle as long as the upper-left
 * corner remains above and to the left of the lower
 * right corner.
 */
public class Rectangle { ... }
```



Remove this page from the exam and use it to answer the following questions. Do not write on this page or include it with the rest of the exam when you turn it in.

CSE 331 18wi Midterm Exam 2/8/18

Question 5. (4 points) Constructor and rep exposure. The representation of a `Rectangle` consists of two `Point` objects giving the corner coordinates. Here are the declarations of those instance variables and the constructor code that initializes them. (We will use this rep in all remaining questions about `Rectangles`.)

```
public class Rectangle {
    private Point ul;    // upper-left corner
    private Point lr;    // lower-right corner

    /**
     * Construct a new Rectangle
     * @effects Make a new rectangle with given corners
     * [other specification details omitted]
     */
    public Rectangle(Point ul, Point lr) {
        this.ul = ul;
        this.lr = lr;
        checkRep();
    }
}
```

Are there any potential representation exposure problems if we have these instance variables and they are initialized in this way by this constructor? (circle)

Yes No

Give a brief (a couple of sentences max) explanation for your answer.

CSE 331 18wi Midterm Exam 2/8/18

Question 6. (12 points) RI, AF, and checkRep. (Hint: these are pretty simple – don't panic if the answers are short.)

(a) (3 points) Give a suitable Representation Invariant (RI) for class `Rectangle`.

(b) (3 points) Give a suitable Abstraction Function (AF) for class `Rectangle`.

(c) (6 points) Complete the implementation of method `checkRep` for class `Rectangle`. This method should verify as much of the rep invariant as is reasonable. If no error is detected, `checkRep` should return quietly without doing anything further.

```
/** Terminate execution with an assertion failure if a
 * violation of the rep invariant is discovered. */
public void checkRep() {
```

```
}
```

CSE 331 18wi Midterm Exam 2/8/18

Question 7. (18 points) Specification and implementation. We would like to add two methods to our `Rectangle` class. For each of these methods, give a proper JavaDoc specification and an implementation of the method. The first line of the JavaDoc comment and the method heading are supplied for you. You need to write the rest of the Javadoc (including `@` tags) and the method code. Hint: don't forget `checkRep`.

If there is more than one reasonable way to write a specification or method, pick the one that seems best. You do not need to explain your choice(s).

(a) (10 points) `contains`. If `r` is a `Rectangle` and `p` is a `Point`, then `r.contains(p)` should return `true` if `p` is inside `r` and `false` if it is not. If `p` lies on the border (an edge or corner) of `r`, then it is considered to be “inside” `r`, and the method should return `true`.

```
/** Test whether a point is contained in this.
 *
 *
 *
 *
 *
 *
 *
 *
 *
 *
 *
 *
 *
 *
 *
 */
public boolean contains(Point p) {
```

```
}
```

CSE 331 18wi Midterm Exam 2/8/18

Question 7. (cont.) (b) (8 points) `setUL`. If `r` is a `Rectangle` and `p` is a `Point`, then `r.setUL(p)` should change the upper-left corner of the rectangle so it has the coordinates given by point `p`. (Hint: the implementation might be very simple, but it is important to provide an appropriate specification. Think about the rep invariant (RI), and abstraction function (AF) for `Rectangle` and be sure the specification covers any unusual cases.)

```
/** Change the upper-left corner of this.
 *
 *
 *
 *
 *
 *
 *
 *
 *
 *
 *
 *
 *
 *
 *
 *
 *
 *
 *
 */
public void setUL(Point p) {
```

```
}
```

CSE 331 18wi Midterm Exam 2/8/18

Question 8. (9 points, 3 each) Testing. Describe three distinct black-box tests that could be used to verify that the `contains` method from the previous problem works properly. Each test description should describe the test input and expected output. For full credit each test should be different in some significant way from the other tests (think about boundary conditions and subdomains, etc.). You do not need to provide JUnit or other code, just be sure you give a clear, precise description of each test.

(a)

(b)

(c)

CSE 331 18wi Midterm Exam 2/8/18

Question 9. (12 points) `equals` and `hashCode`. Below, give implementations of suitable `equals` and `hashCode` methods for class `Rectangle`.

For `equals`, two `Rectangle`s are considered equal if they have the same coordinates, i.e., their corners are equal.

For `hashCode`, your solution should return a good quality `hashCode` value, i.e., while “`return 42;`” does, in fact, meet the requirements for a `hashCode` method, it is not a good implementation.

You do not need to add to or complete the JavaDoc specifications – just write the code.

```
/** Return true if other is a Rectangle with the same
 * coordinates as this. */
@Override
public boolean equals(Object other) {
```

```
}
```

```
/** Return a suitable hash code for this */
@Override
public int hashCode() {
```

```
}
```

CSE 331 18wi Midterm Exam 2/8/18

Final questions, not related to the `Rectangle` or `Point` classes...

Question 10. (12 points, 4 points each part) Specifications and implementations.

Each part of this question gives two specifications and one implementation. For each part, you should (i) indicate which specification is stronger than the other, or indicate they are incomparable if neither one is stronger than the other, and (ii) indicate which of the specifications (one or both or neither) are satisfied by the given implementation.

(a)

Specification S1:

```
@requires x >= 10
@return -1 if x > 10 or 0 otherwise
```

Specification S2:

```
@requires x > 10
@return -1
```

Implementation:

```
public static int m1(int x) {
    int i = 0;
    if (x > 10) {
        i = -1;
    }
    return i;
}
```

(i) Which specification is stronger? (circle) S1 S2 incomparable

(ii) Which specifications are satisfied by the implementation? (circle one, both, or neither as appropriate):

S1 S2

(continued on next page)

CSE 331 18wi Midterm Exam 2/8/18

Question 10. (cont.) (b)

Specification S1:

```
@throws IllegalArgumentException if k <= 0
@return the square of k
```

Specification S2:

```
@return the square of k
```

Implementation:

```
public static int m2(int k) {
    if (k <= 0) {
        throw new IllegalArgumentException();
    }
    return k*k;
}
```

(i) Which specification is stronger? (circle) S1 S2 incomparable

(ii) Which specifications are satisfied by the implementation? (circle one, both, or neither as appropriate):

S1 S2

(c) Specification S1:

```
@requires a!=null && a.size() > 0
@modifies a
@effect remove the last element and all negative
elements of a
```

Specification S2:

```
@requires a!=null && a.size() > 0 && all elements of a
are positive (i.e., > 0)
@modifies a
@effect remove the last element of a
```

Implementation:

```
public static void m3 (List<Integer> a) {
    int sz = a.size()-1;
    for (int k = sz; k >= 0; k--) {
        if (k == sz || a.get(k) < 0) {
            a.remove(k);
        }
    }
}
```

(i) Which specification is stronger? (circle) S1 S2 incomparable

(ii) Which specifications are satisfied by the implementation? (circle one, both, or neither as appropriate):

S1 S2

CSE 331 18wi Midterm Exam 2/8/18

Question 11. (2 free points) (All reasonable answers receive the points. All answers are reasonable as long as there is an answer. 😊)

(a) (1 point) What question were you expecting to appear on this exam that wasn't included?

(b) (1 point) Should we include that question on the final exam? (circle or fill in)

Yes

No

Heck No!!

!@#\$%^*% No !!!!!

No opinion / don't care

None of the above. My answer is _____.

CSE 331 18wi Midterm Exam 2/8/18

Additional space for answers if needed. Please indicate clearly which questions you are answering here, and also be sure to indicate on the original page that the rest of the answer can be found here.