THE UNIVERSITY OF QUEENSLAND AUSTRALIA

This exam paper must not be removed from the venue

Venue	____________________
Seat Number	________
Student Number	|__|__|__|__|__|__|__|__|
Family Name	_____________________
First Name	_____________________

School of Information Technology and Electrical Engineering

EXAMINATION

Semester Two Final Examinations, 2018

CSSE2002 Programming in the Large

This paper is for St Lucia Campus students.

Examination Duration:	120 minutes
Reading Time:	10 minutes

Exam Conditions:

This is a Central Examination

This is a Closed Book Examination – specified materials permitted

During reading time - write only on the rough paper provided

This examination paper will be released to the Library

Materials Permitted In The Exam Venue:

(No electronic aids are permitted e.g. laptops, phones)

Calculators - No calculators permitted

One A4 sheet of handwritten or typed notes double sided is permitted

Materials To Be Supplied To Students:

1 x 14-Page Answer Booklet

Instructions To Students:

Additional exam materials (eg. answer booklets, rough paper) will be provided upon request.

Answer all questions in the answer book provided.

Students must hand in the A4 sheet of notes and all working notes when submitting the examination.

For Examiner Use Only

Question	Mark

Total ________

Question 1: [21 marks total]

(A) [6 marks]

The following code compiles and works correctly but contains several stylistic errors. Fix the code according to the Google Style Guide. **Do not write comments.**

You may assume the class `Gear` exists, and has the method `getValue()`, which returns an `int`.

```
import java.util.*;
public class playercharacter extends java.lang.Object{
String NAME;
ArrayList<Gear> INVENTORY;
public playercharacter(java.lang.String name)
{
    this.NAME = name;
    this.INVENTORY = new java.util.ArrayList<Gear>();
}
public int gettotalworth()
{
    int x = 0;
        for (Gear g :
         INVENTORY) {
x += g.getValue();
} return x;
}
}
```

(B) [15 marks]

Write the code for the class `Gear`, according to the following specification. **Variable comments are encouraged. Method comments are not required.**

```
/** Class Gear: An item of gear carried by a character.  */

/** Constructor: Creates an item of gear with the given type and
value
@param type A String representing the gear’s type.
@require Valid values for type are “weapon”, “treasure”, “potion”
         and “junk”. If an invalid value is provided as an input,
         throw an InvalidGearTypeException.
@param value An int representing the gear’s value.
@require value cannot be negative.
*/

/** getValue()
@return The gear’s value.
*/

/** setValue(): sets the value of the gear to x.
@param x The new value of the item. Requirements from the
        constructor must apply.
*/

/** getType()
@return The gear’s type.
*/

/** setType(): sets the type of the gear to t.
@param t The gear’s new type. Requirements from the constructor
        must apply.
*/
```

You may assume the `InvalidGearTypeException` class exists and extends `Exception`.

Question 2: [12 marks total]

Consider the following code snippet. Line numbers have been provided for easy reference.

```
 1 public static String encode(String message) {
 2     String output = "";
 3     for (int i = 0; i < message.length(); i++) {
 4         if (message.charAt(i) >= 'a' &&
 5             message.charAt(i) <= 'z') {
 6             output += (char) (message.charAt(i) + 5);
 7         } else {
 8             output += message.charAt(i);
 9         }
10     }
11     return output;
12 }
13
14 public static void main(String[] args) {
15     String secretMessage = encode(“Hi! xoxox”);
16     System.out.println(secretMessage);
17 }
```

Reminder: the `charAt(x)` method returns the character at index `x` of a `String`.

Also consider the following ASCII table:

Char	SPACE	!	"	#	$	%	&	'	(	)	*	+
Value	32	33	34	35	36	37	38	39	40	41	42	43
Char	,	-	.	/	0	1	2	3	4	5	6	7
Value	44	45	46	47	48	49	50	51	52	53	54	55
Char	8	9	:	;	<	=	>	?	@	A	B	C
Value	56	57	58	59	60	61	62	63	64	65	66	67
Char	D	E	F	G	H	I	J	K	L	M	N	O
Value	68	69	70	71	72	73	74	75	76	77	78	79
Char	P	Q	R	S	T	U	V	W	X	Y	Z	[
Value	80	81	82	83	84	85	86	87	88	89	90	91
Char	P	Q	R	S	T	U	V	W	X	Y	Z	[
Value	80	81	82	83	84	85	86	87	88	89	90	91
Char	\	]	^	_	`	a	b	c	d	e	f	g
Value	92	93	94	95	96	97	98	99	100	101	102	103
Char	h	i	j	k	l	m	n	o	p	q	r	s
Value	104	105	106	107	108	109	110	111	112	113	114	115
Char	t	u	v	w	x	y	z	{	|	}	~	DEL
Value	116	117	118	119	120	121	122	123	124	125	126	127

(A) [2 marks]

What will be printed by the `main` method?

(B) [1 mark]

Explain why the `encode()` method could not modify the `message` string and return it.

(C) [2 marks]
What is the purpose of the cast in line 6? What would happen if it weren't there?

(D) [6 marks]
Rewrite the `encode()` method so that the parameter `message` is a `char[]`, rather than a `String`. Your return type should be either `void` or `char[]`. The method should still loop through the `char[]` and encode characters according to the conditions provided in the example code.

(E) [1 mark]
In relation to question 2(D) above, explain why `void` could be an appropriate return type if `message` is a `char[]`.

Question 3: [8 marks]

The method `printFactors()` shown below is supposed to print out the factors of x in the range 1..x inclusive.

When writing test cases, **do not** attempt to write Junit tests. Just list the inputs with the expected outputs and justification, e.g.:

Input	**Expected print-out**		**Justification**
x	->	y	because…

Be aware that no marks will be awarded without a good justification.

```
/**@require x > 0
 * @ensure All factors of x are printed.
 *         Each factor will be printed exactly once
 *         (even if it is a square root), one factor per line.
 */
static void printFactors(int x) {
    int i = 1;
    int max = x/2;
    while (i < max) {
        if (x % i == 0) {
            System.out.println(i);
            max = x / i;
            if (max != i) {
              System.out.println(max);
            }
        }
        i++;
    }
}
```

(A) [4 marks]
Write a black-box test suite for the `printFactors()` method.

(B) [3 marks]
Write a white-box test suite for the `printFactors()` method giving **path** coverage.

(C) [1 mark]
This method does not meet the specification. Would one or more of your tests have identified an issue? Which ones?

Question 4: [8 marks total]

Consider the following class definitions.

```
public class A {
    /**@require x != null && c != null && c is a character such
     *          that ‘a’ <= c <= ‘z’
     * @ensure \result is the number of times c appears in x
     */
    public int charCounter(String x, char c) { }
}

public class B extends A {
    /**@require x != null && c != null && c is the character
     *          ‘a’, ‘e’, ‘i', ‘o’ or ‘u’
     * @ensure \result is the number of times c appears in x
     */
    public int charCounter(String x, char c) { }
}

public class C extends A {
    /**@require x != null && c != null && c is a character such
     *          that ‘a’ <= c <= ‘z’ || ‘A’ <= c <= ‘Z’
     * @ensure \result is the number of times c appears in x
     */
    public int charCounter(String x, char c) { }
}
```

Reminder: `\result` in the method specifications refers to the value returned by the method.

(A) [1 mark]
Define the substitution principle.

(B) [1 mark]
What implications does the substitution principle have for designing subclasses?

(C) [3 marks]
Does class B satisfy the substitution principle with respect to class A? Explain why or why not.

(D) [3 marks]
Does class C satisfy the substitution principle with respect to class A? Explain why or why not.

Question 5: [6 marks]

Consider the following code. Line numbers have been provided for easy reference.

```
 1 import java.util.Scanner;
 2
 3 public class FactorCalculator {
 4     public static void main(String[] args) {
 5         Scanner sc = new Scanner(System.in);
 6         System.out.println("Enter a number: >");
 7         int number = Integer.parseInt(sc.next());
 8         int[] factors = getFactors(number);
 9         System.out.println("The factors of " + number +
10                            " are:");
11         for (int i = 0; i < factors.length; i++) {
12             System.out.println(factors[i]);
13         }
14     }
15
16     static int[] getFactors(int x) {
17         int[] factors = new int[x / 2];
18         int elementsInFactors = 0;
19         for (int i = 1; i <= x; i++) {
20             if (isFactor(x,i)) {
21                 factors[elementsInFactors] = i;
22                 elementsInFactors++;
23             }
24         }
25         return factors;
26     }
27
28     static boolean isFactor(int x, int y) {
29         return (x % y) == 0;
30     }
31 }
```

Reminders:

- The `next()` method of a `Scanner` will return the next token from an input stream. The default delimiter is white space.
- `Integer.parseInt()` will try to convert a number represented as a `String` into an `int`.

(A) [3 marks]
Identify a line of code which may generate exceptions. What exception/s could occur on that line? Give an example of a situation in which the exception could occur.

(B) [3 mark]
Rewrite the relevant section of code to handle the exception.

END OF EXAMINATION