ICS 4U0 **Semester 2, 2012 – 2013**

Grade 12 Computer and Information Science
Mid-Unit Quest – Introduction to C# and Review of Programming

Mr. N. Nolfi
Victim: Mr. Solutions Brilliant work Mr. S.!!

KU	APP	TIPS	COM
24/24	20/20	10/10	7/7

1. Match each term in the left column with the ***best*** definition in the right column. **(16 KU)**

U object	A. A tangible and visible entity.
Q braces	B. An appliance that corrects dental irregularities. 
N primitive data type	C. A method of conveying information used by "cave men."
K assignment statement	D. Something students hate to get from their teachers.
P !	E. It's so exciting to be studying computer science!!!!!!!!!!!!!!!!!!!!!!!!!!!!
I %	F. A function or "action" that belongs to an object.
J index	G. Operators, such as &&, || and ! that are used to create compound conditions.
L repetition (looping)	H. Any time during which a program is being executed.
M data field (property)	I. Operator used to evaluate the remainder obtained upon dividing two integers.
R selection ("if")	J. A number that is used to identify a particular character of a string.
F method	K. A statement that is used to give a value to a variable.
G conditional operators	L. A programming structure that allows a particular group of statements to be repeated a certain number of times or while a certain condition is true.
S declaration	M. A variable that belongs to an object.
H run-time	N. A data type that is not defined in terms of simpler types.
O design-time	O. Any time during which a program's source code is being edited.
T compile	P. The "not" operator used in C-style languages.
	Q. Symbols used to enclose a group of statements that are to be treated as a single statement.
	R. A programming structure that allows a particular group of statements to be executed while other groups of statements may be ignored.
	S. A statement that specifies the name, data type and other aspects of a variable.
	T. Translate a high-level program to bytecode, assembly code or machine code.
	U. Generally a collection of properties (data fields) and methods. In the .NET environment, events are also included.

2. Translate into C# assignment statements. **(8 KU)**

(a) Calculate the number of whole ***minutes*** in a given number of seconds.	// Value of 320 assigned for illustration purposes only!
	int seconds = 320;
	int minutes = seconds / 60;

(b) $A = 2\pi r^2 + 2\pi rh$	// Values chosen for illustration purposes only!
	double radius = 7.3, height = 2.9, area;
	area = 2 * Math.PI * Math.Pow(radius, 2)
	+ 2 * Math.PI * radius * height;

3. The code given below is supposed to find the average of all integers between 0 and 1000 that are divisible by 5. Unfortunately, the code was written by a very sloppy programmer who made several syntax, logic and indentation errors. Circle each error and then write corrected code in the provided space. **(5 APP)**

Sloppy Code (Circle all errors)	*Corrected Code*

`sum = 0`
`count = 0`
`for (x=0; x >= 1000; x+=5);`
`{`
`sum=+x`
`count++`
`}`
`average = sum/count;`
	`sum = 0;`
	`count = 0;`
	`for (x = 0; x <= 1000; x+=5)`
	`{`
	`sum += x;`
	`count++;`
	`}`
	`average = sum/count;`
	; for assignment statements

4. For the given code snippet, create a memory map and state the problem that is solved. **(10 APP)**

Code Snippet	*Memory Map*	*Problem that is Solved*

`int nakul=0;`
`string c="";`
`string s="aeiouAEIOU";`
`string a="Kenny Vuong";`
`for (int i=0; i<a.Length; i++)`
`{`
`  c=a.Substring(i,1);`
`  if (s.IndexOf(c)==-1)`
`  nakul ++;`
`}`
	Before entering loop → / After exiting loop →

i	c	nakul
–	""	0
0	"K"	1
1	"e"	1
2	"n"	2
3	"n"	3
4	"y"	4
5	" "	5
6	"V"	6
7	"u"	6
8	"o"	6
9	"n"	7
10	"g"	8
–	"g"	8

By the time the loop has finished executing, the variable "nakul" stores…

the number of characters in the string "a" that are NOT vowels. (The space is considered "NOT a vowel")

5. Consider the following C# code: **(2 APP)**

```
long b = 3;
int a = b;
```

Is this valid C# code? If so, explain why. If not, explain why it isn't and make corrections.

The value of "b" cannot be assigned to "a" because "b" is of type "long," which is too large for the "int" type.

Correction:

```
int a = (int) b;
```

6. Explain how the following C# code could be improved: **(3 APP)**

```
if (smallRadioButton.Checked)
    pizzaSize = "small";
if (mediumRadioButton.Checked)
    pizzaSize = "medium";
if (largeRadioButton.Checked)
    pizzaSize = "large";
if (extraLargeRadioButton.Checked)
    pizzaSize = "extra large";
```

"pizzaSize" can have only one of the 4 possible values. ∴ a single "if" with multiple clauses should be used.

Correction:

```
if (smallRadioButton.Checked)
   pizzaSize = "small";
else if (mediumRadioButton.Checked)
   pizzaSize = "medium";
else if (largeRadioButton.Checked)
   pizzaSize = "large";
else
   pizzaSize = "extra large";
```

7. Two integers are ***relatively prime*** if their only common divisor is ***one***. For example, 9 and 14 are relatively prime because their only common divisor is 1. However, 8 and 14 are ***not*** because 2 divides into both.

(a) Explain the steps that a ***computer*** could execute to determine whether the integers 12 and 35 are relatively prime. **(3 TIPS, 2 COM)**

Strategy — how well it's explained

i	Does i divide into 12?	Does i divide into 35?
2	Yes	No
3	No	No
4	Yes	No
5	No	Yes
6	Yes	No
7	No	Yes
8	No	No
9	No	No
10	No	No
11	No	No
12	Yes	No

- i represents the candidate divisor
- the value of i ranges from 2 to the smaller of the two #'s
- In this example, no value of i is found that divides exactly into both 12 and 35.
- This means that 12 and 35 must be relatively prime

The search ends at 12 because no number greater than 12 can divide into 12

(b) Write a C# program snippet that determines whether two given integers are relatively prime. Your code should include variable declarations. **(7 TIPS, 5 COM)**

For indentation, variable name, etc.

```
int a = 132, b = 27;  // Values chosen for illustration purposes only!
int i = 2, smaller;
if (a < b)
    smaller = a;
else
    smaller = b;
while ((a%i != 0 || b%i != 0) && i <= smaller)
    i++;
if (a%i == 0 && b%i == 0)
    label1.Text = "Not relatively prime.";
else
    label1.Text = "Relatively prime.";
```

Kabilan, Kenny and Raaghav also used this method.

Crysta's Method

If gcd(a,b) = 1 → relatively prime
If gcd(a,b) > 1 → NOT relatively prime

```
// Euclidean gcd algorithm
// r → remainder, i.e. r = a%b
do
{
   r = a%b;
   a = b;
   b = r;
} while (b != 0);
gcd = a;
if (gcd > 1)
   label1.Text = "NOT rel. prime";
else
   label1.Text = "rel. prime";
```