ICS 4U0 | Semester 2, 2012 – 2013

Grade 12 Computer and Information Science
Major Test – Introduction to C# and Review of Programming

Mr. N. Nolfi

Victim: Mr. Solutions

Superb work Master of Solutions!

KU	APP	TIPS	COM
20/20	21/21	13/13	10/10

1. Match each term in the left column with the ***best*** definition in the right column. **(20 KU)**

V static

N primitive data type

F cast

P !

W instance

X strongly-typed

R selection ("if")

L repetition (looping)

M data field / property

H run-time

E method

T compile

S declaration

J index

G conditional operators

K assignment statement

U object

I %

O design-time

Q braces

KU	APP
–0	–0
TIPS	**COM**
–0	–0

~~A~~. A crackling or hissing noise caused by electrical interference.

~~B~~. A method of conveying information used by "cave men."

~~C~~. Bandage consisting of a firm covering (often made of plaster of Paris) that immobilizes broken bones while they heal.

~~D~~. How girls feel whenever Luka and Chase enter the room.

~~E~~. A function or "action" that belongs to an object.

~~F~~. Force a conversion from one data type to another.

~~G~~. Operators, such as &&, || and ! that are used to create compound conditions.

~~H~~. Any time during which a program is being executed.

~~I~~. Used to evaluate the remainder obtained upon dividing two integers.

~~J~~. A number that is used to identify a particular character of a string.

~~K~~. A statement that is used to give a value to a variable.

~~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.

~~M~~. A variable that belongs to an object.

~~N~~. A data type that is not defined in terms of simpler types.

~~O~~. Any time during which a program's source code is being edited.

~~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.

~~V~~. A member of a class that ***can*** be accessed ***without*** creating an object.

~~W~~. A member of a class that ***cannot*** be accessed unless an object is first created.

~~X~~. A language in which all variables must have a defined data type, implicit conversions are generally not allowed and strict type-checking is performed at run-time.

Oh Luka! Oh Chase!

Girl who fainted after Luka and Chase entered the room.

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

Code Snippet

```
int i=-1;
string c1="", c2="";
string a="r??gib??n";
string b="robgibson";
do
{
   i++;
   c1=a.Substring(i,1);
   c2=b.Substring(i,1);
}while ((c1==c2 || c1=="?")
    && i<a.Length-1 && i<b.Length-1);
bool x;
if (i==a.Length-1 && c1==c2)
   x=true;
else
   x=false;
```

Memory Map

Values Before Entering Loop

Values of i

Values After Exiting Loop

i	c1	c2	x
-1	""	""	–
0	"r"	"r"	–
1	"?"	"o"	–
2	"?"	"b"	–
3	"g"	"g"	–
4	"i"	"i"	–
5	"b"	"b"	–
6	"?"	"s"	–
7	"?"	"o"	–
8	"n"	"n"	–
8	"n"	"n"	true

Problem that is Solved

By the time the loop has finished executing, the variable "x" stores… "true" if strings "a" and "b" match and "false" otherwise. The strings match if corresponding characters are identical, except wherever a "?" is found, which matches any character.

3. Consider the following description of the method "Replace" found within the String class. Then answer the questions given below.

`public string` `Replace(`**`string`** `oldValue,` **`string`** `newValue)`

Returns a new string in which ***all*** occurrences of a specified string (oldValue) in the current ***instance*** are replaced with another specified string (newValue).

(a)
```
string a="sauna", b="au", c="k";
a=a.Replace(b,c);
a=a.Replace("a",c);
```

After all the above instructions are executed, what is the value of the string variable "a?" Explain. **(3 APP)**

- initial value of "a" → "sauna"
- after first call of "Replace", the value of "a" changes to "skna" because "au" is replaced with "k"
- after the second call, the value of "a" changes to "sKnk" because "a" is replaced with "k"

(b) Is "Replace" a static or instance method? Explain. **(3 APP)**

"Replace" is an instance method. We know this because...

(a) It is stated in the description

(b) A call to the "Replace" method only makes sense if there is an existing string in which "oldValue" is replaced with "newValue."

(c) After the following code is executed, what is the value of the string variable "b?" Explain. **(3 TIPS)**

```
string a="aeiou";
string b="Crysta Ivancic";
for (int i=0; i<a.Length; i++)
    b=b.Replace(a.Substring(i,1), "*");
```

i	b
0	"Cryst* Iv*ncic"
1	"Cryst* Iv*ncic"
2	"Cryst* Iv*nc*c"
3	"Cryst* Iv*nc*c"
4	"Cryst* Iv*nc*c"

The final value of "b" is "Cryst* Iv*nc*c" because this program snippet replaces all occurrences of lowercase vowels with a "*". (The uppercase "I" is NOT replaced because it is not found within the string "a")

KU	APP
– 0	– 0
TIPS	**COM**
– 0	– 0

4. Write a C# program segment that calculates the checksum of a given credit card number. (For the sake of consistency and also for your convenience, some variables are already declared for you and have been given values. Use these variable names in your code!) **(5 APP, 5 COM)**

This problem can be solved using a variety of different approaches.

```
string cardNumber=cardNumberTextBox.Text;
int length=cardNumber.Length;
int checksum=0;
string reversedCardNumber ="";
// Build a string that consists of exactly the same digits as "cardNumber" but in the opposite order
for (int i =0; i<length; i++)
    reversedCardNumber = cardNumber.Substring(i,1)+ reversedCardNumber;
// Calculate checksum
for (int i=0; i< length; i++)
{
    int digit = Convert.ToInt32(reversedCardNumber.Substring(i,1));
    if (i%2 ==1)
    {
     digit *=2;
     if (digit >9)
        digit -=9;
    }
    checksum+=digit;
}//end of for
```

Marks allotted according to approach used.

5 → works, no errors
4 → minor errors
3 → on the right track, more errors
2 → vaguely on the right track, many errors
1 → wrote some C# code, somewhat relevant, mostly off track
0 → blank or irrelevant C# code (WTF?)

5. An integer is called ***perfect*** if the sum of its ***proper divisors*** is equal to the number itself. Two examples of perfect numbers are 6 and 28 because 6 = 1 + 2 + 3 and 28 = 1 + 2 + 4 + 7 + 14. (Note that a proper divisor of a number is any divisor of the number that is smaller than the number.)

Write a C# program segment that finds all perfect numbers less than or equal to 10000. **(10 TIPS, 5 COM)**

```
int sumOfProperDivisors =1, number = Convert.ToInt32(numberTextBox.Text);
for (int i=2; i<=number/2 ; i++)
{
    if ( number % i == 0)
       SumOfProperDivisors += i;
}

if  (sumOfProperDivisors == number)
    outputLabel.Text = "Perfect";
else
    outputLabel.Text= "Not Perfect";
```

COM Marks Given For...

- correct indentation
- use of descriptive identifier names
- overall clarity of code
- correct syntax (e.g. correct use of semi-colons)

KU	APP
–0	–0
TIPS	COM
–0	–0