American University in Cairo
Department of Computer Science & Engineering
CSCE 106

Instructor Name: Dr. Khalil

Last Name: .......................................................... ID:..............................
First Name: .......................................................... SECTION

* Do not turn this page until asked to do so.
* Exam time is 120 minutes.
* Put the answers on the same question sheet; do not use any additional papers, even for scratch.
* Write your name, ID, section no. in the indicated places.
* Read the exam instructions.
* Read the Academic Integrity policy.
* Sign the following statement.

Academic Integrity POLICY

Cheating in Exams is a violation of the Academic Integrity policy of AUC. Whispering, talking, looking at someone else’s paper, or copying from any source is considered cheating. Any one who does any of these actions or her/his answers indicates that she/he did any of them, will receive a punishment ranging from zero in this exam to failing the course. If repeated, it may lead to dismissal from AUC.

I have read the honesty policy and exam instructions and I am presenting this exam as entirely my effort.

Signature: ____________

DO NOT USE THIS SECTION

<table>
<thead>
<tr>
<th>Question</th>
<th>Points</th>
<th>Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>25</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>100</td>
<td></td>
</tr>
</tbody>
</table>
Question 1 (15 points)
Tick only one possible answer for each of the following:

1) The value of $(x - 4 != 4) \&\& (x - 4 == 4)$ is:
   a. $x - 4$ [ ]
   b. True [ ]
   c. $x + 4$ [ ]
   d. None of the above [ ]

2) $(101000111)_{2}$ is equivalent to:
   a. $(507)_{8}$ [ ]
   b. $(507)_{16}$ [ ]
   c. $(5544)_{10}$ [ ]
   d. None of the above [ ]

3) $(-15)_{10}$ in 8-bit two's complement format is:
   a. 110001 [ ]
   b. 11110000 [ ]
   c. 11110001 [ ]
   d. None of the above [ ]

4) Von Neumann architecture is:
   a. A stored-program computer [ ]
   b. A web browser [ ]
   c. An architecture for RAM [ ]
   d. All the above [ ]

5) The value of $10 \% 4 * 3 - 8 >= 18 + 30 / 4 - 20$ is:
   a. -2 [ ]
   b. True [ ]
   c. False [ ]
   d. None of the above [ ]

6) Software Piracy is:
   a. Violation of copyright agreements by illegally copying software [ ]
   b. Code disrupting the computer operations and erasing information [ ]
   c. A Computer theft or computer fraud [ ]
   d. A biological disease causing malfunctioning of the computer [ ]

7) The RAM is used to:
   a. To store fixed code to start up the computer [ ]
   b. To store programs and data that are to be executed [ ]
   c. To perform the arithmetic and logical operations [ ]
   d. None of the above [ ]

8) In binary: $1011 + 11011 = ?$
   a. 101000 [ ]
   b. 100110 [ ]
   c. 110110 [ ]
   d. 100101 [ ]

9) The ALU is the unit of computer responsible of:
   a. Storing programs and data permanently [ ]
   b. Performing arithmetic and logical operations [ ]
   c. Performing Input/Output operations [ ]
   d. None of the above [ ]

10) The Compiler is a software that:
    a. translates a high-level language program into machine language [ ]
    b. translates an assembly language program into machine language [ ]
    c. lets users display and view a web document [ ]
    d. None of the above [ ]
**Question 2 (10 points)**
Write a program fragment that uses nested loops to produce the following output:

```
1 2 3 4 5 6
0 2 3 4 5 6
0 0 3 4 5 6
0 0 0 4 5 6
0 0 0 0 5 6
0 0 0 0 0 6
```
### Question 3 (25 points)
Show the output of the following program segments:

<table>
<thead>
<tr>
<th>Program – 1</th>
</tr>
</thead>
</table>
| int x = 1, y = 1;  
while (x < 4)  
{  
y = y * x++;  
cout << "x = " << setw(3) << x << " y = " << setw(3) << y << endl;  
} |

<table>
<thead>
<tr>
<th>Program - 2</th>
</tr>
</thead>
</table>
| for (int k = 1; k < 30; k++)  
If ((k%5 == 0) || (k%7 == 0))  
cout << setw(3) << k+1;  
cout << endl; |

<table>
<thead>
<tr>
<th>Program - 2</th>
</tr>
</thead>
</table>
| int F[6] = {1, 1};  
const int zero = 0;  
const int one = 1;  
cout << setw(3) << zero << setw(3) << F[0] << endl;  
cout << setw(3) << one << setw(3) << F[1] << endl;  
for (int k = 2; k < 6; k++)  
{  
F[k] = F[k-1] + F[k-2];  
cout << setw(3) << k << setw(3) << F[k] << endl;  
} |

<table>
<thead>
<tr>
<th>Program - 3</th>
</tr>
</thead>
</table>
| int B[4];  
const int two = 2;  
int n = 13;  
int c = 3;  
do  
{  
B[c] = n % two;  
n /= two;  
c--;  
} while (n != 0);  
cout << "The equivalent value of " << n << " is " << c;  
for (int c = 0; c < 4; c++)  
cout << B[c];  
cout << endl; |
Program - 4

```cpp
void nmx ( int, int, float&);    

void main() {
    int n = 3;
    int m = 2;
    float nm = 2.757;
    nmx (n, m, nm);
    cout << "n = " << setw(3) << n << endl;
    cout << "m = " << setw(3) << m << endl;
    cout << setprecision(2);
    cout << "result is:   " << setw(6) << nm << endl;
}

void nmx (int a, int b, float& x) {
    a /= b;
    if (a % b != 0)
        x = 4.75 + a;
    else
        x = 4.75 - b;
}
```

Program - 5

```cpp
for (int k = 7; k <= 16; k++)
    switch (k % 10) {
    case 0: cout << "", ";
            break;
    case 1: cout << "OFTEN ";
            break;
    case 2:
    case 8: cout << "IS ";
            break;
    case 3: cout << "NOT ";
            break;
    case 4:
    case 9: cout << "DONE ";
            break;
    case 5: cout << "WELL";
            break;
    case 6: cout << ".";
            break;
    case 7: cout << "WHAT ";
            break;
    default: cout << "Bad number. ";
    }
    cout << endl;
```
**Question 4 (15 points)**

If the sum of digits of any positive integer number is divisible by 9, the number itself is also divisible by 9.

The following C++ program takes a positive integer number less than 88888 and checks whether it is divisible by 9 or not, using the above mentioned technique. There are some missings (represented by dots) in the given program. Complete these missings such that the program could be compiled and run correctly.

**The Program**

```cpp
#include <iostream>
using namespace std;

void main ()
{
    int num, d, sum;
    ......................;

do
{
    cout << “Enter a positive integer number less than 88888” << endl;
    cin >> num;
} while (..........................);

while (..........................)
{
    d = ..........................;

    sum = sum + d;

    num = ..........................;
}

if (..........................)
    cout << “The number is divisible by 9” << endl;
else
    cout << “the number is NOT divisible by 9” << endl;
}
```
Question 5 (15 points)

Draw a Flow chart and write a complete C++ program that asks the user to enter an integer number greater than or equal to 0 and less than 1000. The program has to check the entered number and if it is less than 10, write out the words **ONE DIGIT BIG!**, if it is between 10 and 99, write out the words **TWO DIGITS BIG!** and if the user's number is greater than 99, write out the phrase **THREE DIGITS BIG!**.

*Add to your program the necessary code to validate the user input. Be sure to use proper formatting and appropriate comments in your code. The output should be clearly labeled. Show the three phases of software development: the analysis, design (draw a Flow Chart), and implementation.*
**Question 6 (20 points)**

A perfect number is a positive integer having the property that the sum of its factors (including 1) equal to the number itself. The first perfect number is 6 because its factors are 1, 2 and 3 and their sum is 6 which is equal to the number itself. The next perfect number is 28 because its factors are 1, 2, 4, 7 and 14 and their sum is 28 which is equal to the number itself. Write a modular C++ program that takes a positive integer number greater than or equal to 0 and less than 9999, checks whether the number is perfect or not, and prints out an appropriate message indicating the case. The program should repeat to take another number to check it and continue to repeat until a value 0 is entered for the number. (*Hint: Define a function perfect that takes a number, checks it and returns an appropriate output indicating the case.*) Enforce validation on the user input such that the program should not accept only integer numbers in the given range.

**The program**

```

Good Luck

```