

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

Instructor: Dr. Khalil	Final Exam	Spring 2012
Last Name :		ID:
First Name: Section No.:		

EXAMINATION INSTRUCTIONS

- * 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 honesty policy.
- * Sign the following statement.

Academic Integrity Policy

Cheating in Exams is a violation of the honesty 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

Question	Points	Grade
1	10	
2	25	
3	15	
4	10	
5	15	
6	25	
Total	100	

Question 1 (10 points)

Tick only one possible answer for each of the following:

- 1) The Boolean expression $\neg(x < y)$ can be simplified into:
- a. $(x > y)$ ☐
 - b. $(x \leq y)$ ☐
 - c. True ☐
 - d. none of the above ☐
- 2) **Intellectual Property:**
- a. A form of protection that can be granted by the government for an invention; gives exclusive rights of an invention to its inventor for 20 years. ☐
 - b. Standards of moral conduct as they relate to computer use. ☐
 - c. The rights to which creators of original creative works (such as software, artistic, literary works, inventions, and more) are entitled. ☐
 - d. The legal right to sell, publish, or distribute an original software, artistic or literary work; is held by the creator of a work as soon as it exists in physical form. ☐
- 3) The Boolean expression $((x == 4) \&\& ((x != 4))$ can be simplified into:
- a. $(x - 4)$ ☐
 - b. True ☐
 - c. $(x + 4)$ ☐
 - d. none of the above ☐
- 4) 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 ☐
- 5) The boolean expression $((x \leq y) \parallel \text{False})$ can be simplified into:
- a. $(x > y)$ ☐
 - b. $(x \leq y)$ ☐
 - c. False ☐
 - d. none of the above ☐
- 6) 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 ☐
- 7) In binary: $1111 + 1111 = ?$
- a. 10000 ☐
 - b. 11110 ☐
 - c. 11011 ☐
 - d. 11111 ☐
- 8) In C++, the statement: `if (1 == 1) cout << 1; else cout << 0;` will produce:
- b. 1 ☐
 - c. $(1 != 1)$ ☐
 - d. 0 ☐
 - d. none of the above ☐
- 9) $(10000000)_2$ is equivalent to:
- a. $(256)_{10}$ ☐
 - b. $(127)_{10}$ ☐
 - c. $(128)_{10}$ ☐
 - d. None of the above ☐
- 10) The boolean expression $((x \leq y) \parallel (x == x))$ can be simplified into:
- c. $(x > y)$ ☐
 - d. $(x \leq y)$ ☐
 - e. False ☐
 - d. none of the above ☐

Question 2 (25 points)

Show the output of each of the following program segments:

<pre>void drawLine(char); int func(int); void main () { drawLine('#'); cout << setw(3) << "k" << setw(8) << "func(k)" << endl; drawLine('_'); for (int k=0; k <= 5; k += 2) cout << setw(3) << k << setw(8) << func(k) << endl; drawLine('#'); } int func(int n) { int f = 1; for (int m=2; m <= n; m++) f = m * f; return f; } void drawLine(char ss) { for (int c = 1; c <= 11; c++) cout << ss; cout << endl; }</pre>	
---	--

<pre>float a = 2, b = 2, c = 1.5, x; int d = 4, e = 5.5, y; x = a * b - c + d / e; y = a * (b - c) + e / d; cout << "x = " << setw(4) << x << " y = " << setw(4) << y << endl;</pre>	
---	--

<pre>int B[4], n = 13, c = 3; int nn = n; do { B[c] = nn % 2; nn = nn / 2; c--; } while (nn != 0); cout << "The equivalent value of " << n << " is "; for (int k = 0; k < 4; k++) cout << B[k]; cout << endl;</pre>	
--	--

```

int myNum (int);
void main()
{
    for ( int n = 1; n < 5; n++)
    {
        for (int m = 1; m <= n; m++)
            cout << myNum(n);
        cout << endl;
    }
}
int myNum (int nn)
{
    switch(nn % 3)
    {
        case 1:
            return 1;
            break;
        case 2:
            return 2;
            break;
        default:
            return 3;
    }
}

```

```

void swap (int&, int&);
void main()
{ int a = 55, b = 77, c = 99;
  do
  {
      if ( b > a)
          swap (a, b);
      if ( c > b )
          swap (b, c);
  } while (( b > a ) || ( c > b ));
  cout << "The two required values are:" << endl;
  cout << setw(6) << a << setw(6) << b << endl;
}
void swap (int& X, int& Y)
{ int T = X;
  X = Y;
  Y = T; }

```

```

int x = 1, y = 1, z;
y *= x++;
cout << "x = " << setw(2) << x << "y = " <<
setw(2) << y << endl;
x = 6;
y = 2;
z = --x / y++;
y = z-- + x++;
cout << "x = " << x << " y = " << y << " z = " << z
<< endl;

```

Question 3 (15 points)

Write a C++ program that uses nested loops with selective structures to produce the following output:

```
@
#  #
@ & @
#  #  #  #
@ & @ & @
#  #  #  #  #  #
@ & @ & @ & @
#  #  #  #  #  #  #  #
@ & @ & @ & @ & @
```

Question 4 (10 points)

Draw a Flow Chart for the following C++ main function:

```
void main()
{
    float income, taxRate, taxDue;
    cin >> income;
    while ( income < 0 )
    {
        cout << "Invalid Input, Re-enter!!";
        cin >> income;
    }
    if ( income < 3000 )
        taxRate = 0;
    else
        if ( income < 10000 )
            taxRate = 0.10;
        else
            if ( income < 20000 )
                taxRate = 0.15;
            else
                taxRate = 0.20;
    taxDue = income * taxRate;
    cout << "The Due Tax = " << taxDue << endl;
}
```

Flow Chart

Question 5 (15 points)

A Perfect integer is a positive integer number greater than 1 and the sum of its factors (including 1) is equal to the number itself. For example, 6 is the first perfect number because $1 + 2 + 3$ (the factors of 6) = 6) and 28 is the next perfect number because the sum of its factors $1 + 2 + 4 + 7 + 14 = 28$ which is equal to the number itself.

The following C++ modular program takes a positive integer number greater than 1 and less than 1000 and prompts the user whether the entered number is perfect or not. 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 does not accept any integer number outside the given range (greater than 1 and less than 1000).

The Program

```
# include <iostream>
# include <iomanip>
using namespace std;
bool perfect(int);

void main ()
{   int num;

    cout << "Enter an integer number greater than 1 and less than 1000:" << endl;
    cin >> num;
    while (.....)
    {   cout << "Invalid number!! Please, Re-enter:";
        cin >> num; }

    If ( ..... )
        cout << "The number" << setw(5) << num << "is Perfect" << endl;
    else
        cout << "The number" << setw(5) << num << "is NOT Perfect" << endl;
}

bool ..... (..... n)
{   int sum = .....;

    for ( int c = 2; ..... )
        If ( n % c == 0 )
            .....;

    return (.....);
}
```

Question 6 (25 points)

Write a **modular** C++ program using functions to generate a table showing the area of base and volume of a cone for a sequence of radius and height values (both are integer values). The area of base and volume of a cone are calculated as follows:

$$\text{Area of base} = \pi \times r^2$$

$$\text{Volume of cone} = 1 / 3 \times \pi \times r^2 \times h$$
 where **r** is the radius of base and **h** is the height of the cone,

The program starts both the radius value and the height value by 1 and reads an end value for radius (should be greater than 1 and not exceeding 10) and an end value for height (should be greater than 1 and not exceeding 5), and increments both the radius value and the height value by 1. The program should enforce the necessary validations on both the end radius and end height values.

Both the computed area of base and volume should be printed rounded to the **nearest hundredth**.

(**Hint**: define a function to print the header of the table, a second function takes the value of radius and the value of height and returns the computed area of base and volume of cone (Let it be named **ComputeAreaVolume**), and a third function to print the footer of the table)

An Example: start radius (**startRad**) = 1, end radius (**endRad**) = 8, step radius Value (**stepRad**) = 1, start height (**startHgt**) = 1, end height (**endHgt**) = 3, step height value (**stepHgt**) = 1.

Radius	Height	Area of Base	Volume of Cone
1	1	3.14	1.05
1	2	3.14	2.10
1	3	3.14	3.15
2	1	12.57	4.19
2	2	12.57	8.38
2	3	12.57	12.57
....			
8	3	201.14	201.14

This image shows a single sheet of white paper with horizontal ruling lines. The lines are evenly spaced and run across the width of the page. There are no margins, text, or other markings on the paper.