Assignment 3

Student Name	
Year/Group	2009/2010

Assignment title:	**Introduction to variables**

Unit title:	Computer Skills II (C Language)	**Subject Tutor:**	mhd. Mazen al-Mustafa
Start:		**Coordinator**	mhd. Mazen al-Mustafa
Submission:			
Actual Submission date:		**Grading.**	
Learning Outcomes covered:			
	• **While Loop** • **Do While Loop** • **for Loop**		
Resources:			
	• C How To Program fifth Edition, by H.M. Deitel & P.J. Deitel, • Teacher's handouts • internet		

True/False

1. The loop repetition condition of a *for statement* is tested at the end of each pass. [False]

2. The body of a while statement must cause the loop repetition condition to become false after a finite number of passes to prevent an infinite loop. [True]

3. In counting loops, the counter must be initialized to zero before execution of the loop body begins.
   [False]

4. The loop repetition condition of a while or for statement can be false before the loop begins to execute. [True]

5. Loop counter variables are usually of type double. [False]

6. The body of a *for statement* might not be executed at all. [True]

7. Tables are always displayed using counter-controlled while loops. [False]

8. The update expression in a sentinel-controlled for loop usually involves a call to scanf
   [True]

9. The sentinel value is always the last value added to a sum being accumulated in a sentinel-controlled loop. [False]

10. The loop that follows displays 29 lines of output.

```
for  (i = 0;  i < 30;  i = i + 1)
    printf("%d\n", i);
```

[False]

Multiple Choice

Questions 1-4 refer to the following program segment. Assume that all variables are of type int.

```
z = 0;
g = 0;
s = 0;
i = 0;
while (i < 50) {
    scanf("%d", &t);
    s = s + t;
    if (t >= 0)
```

```
            g = g + 1;
        else
            z = z + 1;
        i = i + 1;
    }
```

1. How many times is the loop body of the while statement executed?

 a. once
 b. never
 c. 49 times
*d. 50 times
 e. until a number 50 or larger is entered

2. The value stored in variable s at the end of the execution of the loop could best be described as ___________.

 a. the average of the numbers scanned
*b. the sum of the numbers scanned
 c. the largest of the numbers scanned
 d. how many numbers were scanned
 e. the sentinel value

3. The value stored in variable z at the end of the execution of the loop could best be described as ___________.

 a. how many positive items were scanned
 b. the sum of all positive items scanned
*c. how many negative items were scanned
 d. the sum of all negative items scanned
 e. the sentinel value

4. The loop can best be categorized as a ___________.

*a. counter-controlled loop
 b. sentinel-controlled loop
 c. loop that computes a product
 d. general conditional loop
 e. none of the above

5. A special value that marks the end of a list of input data is called a ___________.

 a. terminal value
*b. sentinel value
 c. loop control value

d. input value
e. loop termination value

6. A C programmer should not use an expression like the one that follows because _______.

```
n = i + x * ++i;
```

*a. the expression may produce different results on different ANSI C compilers
b. it is illegal to have two consecutive operators
c. the ++ operator cannot be used in combination with other arithmetic operators
d. the increment operator cannot be used in the same expression as the addition operator
e. none of the above

7. The expression

```
x *= i + j / y;
```

is equivalent to _______.

a. `x = x * i + j / y;`
b. `x = (x * i) + j / y;`
c. `x = (x * i + j) / y;`
*d. `x = x * (i + j / y);`
e. none of the above

8. In a program that uses an endfile-controlled loop to compute product of a list of numeric input data, the product variable must be initialized to ___________.

a. 0
*b. 1
c. -1
d. the EOF value
e. none of the above

9. How many lines of output will be displayed by the following program fragment?

```
i = 0
do {
   for  (j = 0;  j < 4;  j = j + 1)
      printf("%d\n", i + j);
   i = i + 1;
} while (i < 5);
```

a. 0
b. 7
c. 9
d. 16
*e. 20

10. How many lines of output will be displayed by the following program fragment?

```
for  (i = 0;  i < 5;  i = i + 1)
   for  (j = 0;  j < i;  j = j + 1)
      printf("%d  %d\n", i, j);
```

a. 0
b. 5
c. 9
*d. 10
e. 20

Short Answer

1. Loops that terminate upon scanning a special input value are called [sentinel]-controlled loops.

2. What is displayed by the following program segment if all variables are of type int and the input data is the data shown?

```
st = scanf("%d%d%d", &n, &m, &p);
printf("n = %d  m = %d  st = %d\n", n, m, st);
```

Input data: `10  12  hello`

[Answer: `n = 10  m = 12  st = 2`]

3. The intention of the following program fragment is to display the positive integers from 1 through n, but it doesn't work. Correct the while statement so the fragment achieves the desired effect.

```
i = 1;
while (i <= n)
    printf("%d  ", n);
    i = i + 1;
printf("\n");
```

[Answer:

```
    while (i <= n) {
        printf("%d  ", i);
        i = i + 1;
    }
```

]

4. Rewrite the following code segment as an equivalent segment that uses a for statement.

```
    product = 1;
    next = 1;
    while (next <= m) {
        product = product * next;
        next = next + 1;
    }
```

[Answer:

```
    product = 1;
    for  (next = 1;  next <= m;  next = next + 1)
        product = product * next;
```

]

5. Write a do-while loop that repeatedly scans integer values until a positive even number is input. You may assume the input data is a list of integers separated by blanks, and that there is a positive even number in the list.

[Answer:

```
        do
            scanf("%d", &num);
        while (num <= 0 || num % 2 != 0);
```

]

6. What is the output of this code segment?

```
for  (k = 5;  k > 0;  k = k - 1) {
   for  (i = 1; i <= 5 - k;  i = i + 1)
      printf(".");
   for  (j = 1;  j <= 2 * k - 1;  j = j + 1)
      printf("B");
   printf("\n");
}
```

[Answer:

```
BBBBBBBBB
.BBBBBBB
..BBBBB
...BBB
....B
```

]

7. Write an expression to represent the following condition:

x is either equal to 10 or not greater than 5

[Answer: `x == 10 || !(x > 5)`]

8. Write a sequence of two statements that is equivalent to the following three-statement sequence. Use operators with side effects and avoid parentheses where possible.

```
x = y * 10;
y = y + 1;
z = z * (y + 5);
```

[Answer:

```
x = y++ * 10;
z *= y + 5;
```

]

9. What is displayed by the following code fragment if all the variables are of type int?

k = 0;

```
        m = 0;
        for  (p = 0;  p < 10;  p = p + k) {
           k = k + 1;
           m = m + p;
           printf("%4d%4d%4d\n", p, k, m);
        }
```

[Answer:

```
    0  1  0
    1  2  1
    3  3  4
    6  4  10
```

]

10. Which variable in the code segment for question 9 is the loop control variable?

[Answer: p]

Test your self

1. Rewrite the following code fragment so that it uses a "`do...while...`" loop to accomplish the same task.

```
#include "stdafx.h"
int main()
{int n;
printf("Enter a non-negative integer: ");
scanf("%d", &n);
while (n < 0)
{
  printf("The integer you entered is negative.");
  printf("Enter a non-negative integer: ");
  scanf("%d", &n);
}
return 0;}
```

2. Suppose that the following code fragment is executed.

```
#include "stdafx.h"
int main()

{
     const int LENGTH = 21;
     char message[LENGTH];
     printf("Enter a sentence on the line below.\n");
     int i = 0;
     do
     {
          scanf("%c", &message[i]);
          ++i;
     }
     while (i < LENGTH - 1 && message[i] != '\0');
     message[i] = '\0'; // Terminate string with NUL char.
     printf("%s",message);
     return 0;
}
```

Suppose that in response to the prompt, the interactive user types the following line and presses Enter:

```
Please go away.
```

What will the *output* of the code fragment look like?

b. Suppose that the statement " `scanf("%c", &message[i]);`" is replaced by the statement `message[i]=getchar();`

Now what will the *output* of the code fragment look like if, in response to the prompt, the interactive user types the following line and presses Enter?

```
Please go away.
```

3. The loop shown below has been written by an inexperienced C programmer. The behavior of the loop is not correctly represented by the formatting.

```
#include "stdafx.h"
int main()
{int n = 10;
while (n > 0)
n /= 2;
printf( "%d",n * n );

     return 0;}
```

a. What is the output of the loop as it is written?

b. Correct the syntax of the loop so that the *logic* of the corrected loop corresponds to the *formatting* of the original loop. What is the output of the corrected loop?

c. Correct the formatting of the (original) loop so that the new format reflects the logical behavior of the original loop.

4. Rewrite the following code fragment so that it uses a "`do...while...`" loop to accomplish the same task.

```
#include "stdafx.h"
int main()

{int n;
printf("Enter a non-negative integer: \n");
scanf("%d",&n);

while (n < 0)
{
     printf("The integer you entered is negative.\n");
     printf( "Enter a non-negative integer: ");
     scanf("%d",&n);
}
     return 0;
}
```

5. What is the output when the following code fragment is executed?

```
#include "stdafx.h"
int main()

{int n;
float x = 3.8;
n = int(x);
printf( "n = %d\n",n );

return 0;
```

6. What is the output when the following code fragment is executed? Rewrite the fragment to obtain an equivalent code fragment in which the body of the loop is a simple statement instead of a compound statement.

```
#include "stdafx.h"
int main()

{int i = 5;
while (i > 0)
{
  --i;
  printf("%d\n", i);
}

return 0;
}
```

7. The following loop is an *endless* loop: when executed it will never terminate. What modification can be made in the code to produce the desired output?

```
#include "stdafx.h"
int main()

{
printf("Here's a list of the ASCII values of all the upper");
printf(" case letters.\n");
 char letter = 'A';
 while (letter <= 'Z')
   printf("%c %d\n",letter ,int(letter));

return 0;
}
```

8. What is the output when the following code fragment is executed?

```
#include "stdafx.h"
int main()

{printf("Here's a list of the ASCII values of all the upper");
printf(" case letters.\n");
float counter = 0.0;
while (counter <= 1.2){
```

```
printf("%f %d\n",counter ,int(counter));
counter+= 0.2;
}
return 0; }
```

9. What is the output when the following code fragment is executed?

```
#include "stdafx.h"
int main()
{int i;
for ( i= 0 ;i<= 8;i++)
     i=i*i;

printf("%d\n",i);
return 0;}
```

10. Write a program that prints the following pattern,

Use for loops to generate the pattern. All asterisks (*) should be printed by a **single** printf statement of the form *printf (“*”);* (this causes the asterisks to print side by side).

```
*
**
***
****
*****
```

11. Write a program that prints the following pattern,

Use for loops to generate the pattern. All asterisks (*) should be printed by a **single** printf statement of the form *printf (“*”);* (this causes the asterisks to print side by side).

```
*****
 ****
  ***
   **
    *
```

12. Write a program that prints the following pattern,

Use for loops to generate the pattern. All asterisks (*) should be printed by a **single** printf statement of the form *printf (“*”);* (this causes the asterisks to print side by side).

```
    *
   **
  ***
 ****
*****
```

13. Write a program that prints the following pattern,

Use for loops to generate the pattern. All asterisks (*) should be printed by a **single** printf statement of the form *printf (“*”);* (this causes the asterisks to print side by side).

```
*****
****
***
**
*
```

14. Write a program that prints the following pattern,
Use for loops to generate the pattern. All asterisks (*) should be printed by a **single** printf statement of the form *printf (“*”);* (this causes the asterisks to print side by side).

```
    *
   ***
  *****
 *******
*********
 *******
  *****
   ***
    *
```

15. Write a program that prints the following pattern,
Use for loops to generate the pattern. All asterisks (*) should be printed by a **single** printf statement of the form *printf (“*”);* (this causes the asterisks to print side by side).

```
    *****
   *****
  *****
 *****
*****
```

16. Write a program that prints the following pattern,
Use for loops to generate the pattern. All asterisks (*) should be printed by a **single** printf statement of the form *printf (“*”);* (this causes the asterisks to print side by side).

```
*****
 *****
  *****
   *****
    *****
```

17. Write a program that prints the following pattern,
Use for loops to generate the pattern. All asterisks (*) should be printed by a **single** printf statement of the form *printf (“*”);* (this causes the asterisks to print side by side).

```
    *
   **
  ***
 ****
*****
 ****
  ***
   **
    *
```

18. Write a program that prints the following pattern,
Use for loops to generate the pattern. All asterisks (*) should be printed by a **single** printf statement of the form *printf (“*”);* (this causes the asterisks to print side by side).

```
*
**
***
****
*****
****
***
**
*
```

19. Write a program that prints the following pattern,
Use for loops to generate the pattern. All asterisks (*) should be printed by a **single** printf statement of the form *printf (“*”);* (this causes the asterisks to print side by side).

```
  *****
 *******
*********
 *******
  *****
```

20. Write a program that: Print primes numbers from 101 to 200, except 113 number

21. Write a program that:

1- Prompt user to enter 10 salaries
2- Read 10 salaries
3- Add 10 percentage for each salary

4- Print the new salaries

Note: use array to read and write salary

22. Write a program that:

1- Prompt user to enter 10 grades
2- Read 10 grades
3- Print the max

Note: you have to use array to read grades

Student Declaration: I certify that the work contained in this assignment was researched and prepared by me:

Signature: ______________________________ Date: / /

Remark: separate feedback sheet will be returned to you after your work has been marked

International University for Science & Technology (IUST)

College of Information Technology

Feedback Sheet

Student's Notes:

Assistant's Feed Back:

Assistant: ________________________________

Signature: ________________________________ Date: _________________________