Programming Using C++

Lecture 12

Prepared by: Dr. Yasmine Afify

Problem: Cheaper by the Dozen ☺

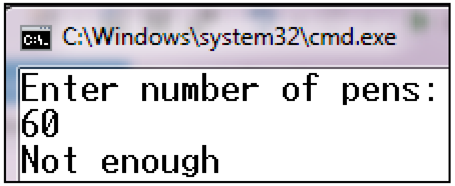
- Implement a program to convert a number to dozens. In case the resulting dozens are 6 or more, the program displays an okay message. Otherwise, it displays that this is not enough.

- Sample Run:

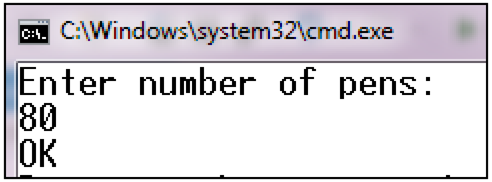
```
Enter number of pens: 40
Not enough
```

Cheaper by the Dozen ☺

```
#include <iostream>
using namespace std;
int main ()
{
   int numPens;
   cout << "Enter number of pens:\n";
   cin >> numPens;
   (numPens/12 >= 6) ? cout<<"OK\n"  :  cout<<"Not enough\n";
   return 0;
}
```



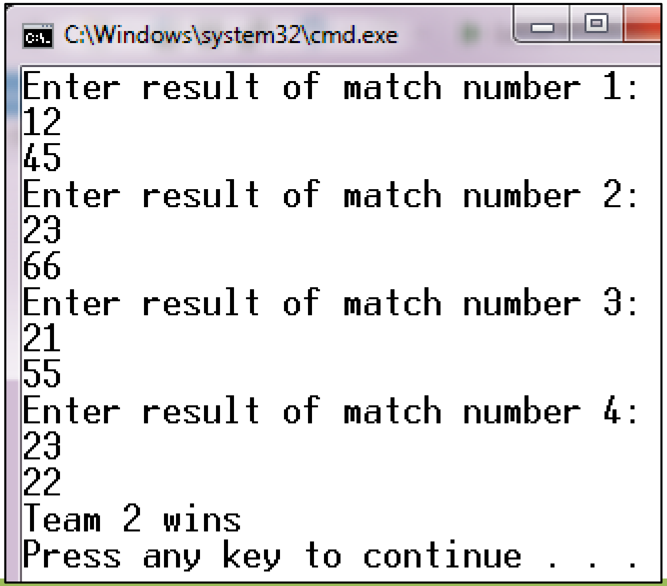
```
Enter number of pens:
60
Not enough
```



```
Enter number of pens:
80
OK
```

Problem: Basketball Championship

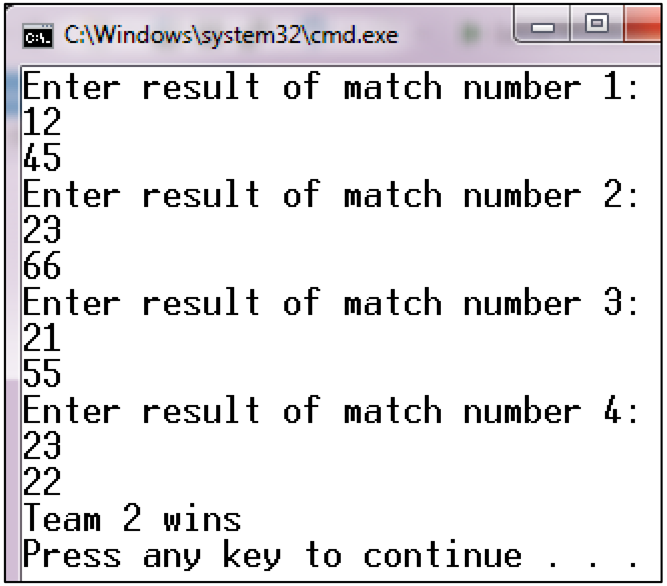
- Program reads result of 4 basketball matches (number of goals scored by two teams).
- For each match, the winner team gets 3 points. In case of tie, each team gets 1 point. The program should display the winner team.



```
Enter result of match number 1:
12
45
Enter result of match number 2:
23
66
Enter result of match number 3:
21
55
Enter result of match number 4:
23
22
Team 2 wins
Press any key to continue . . .
```

Basketball Championship

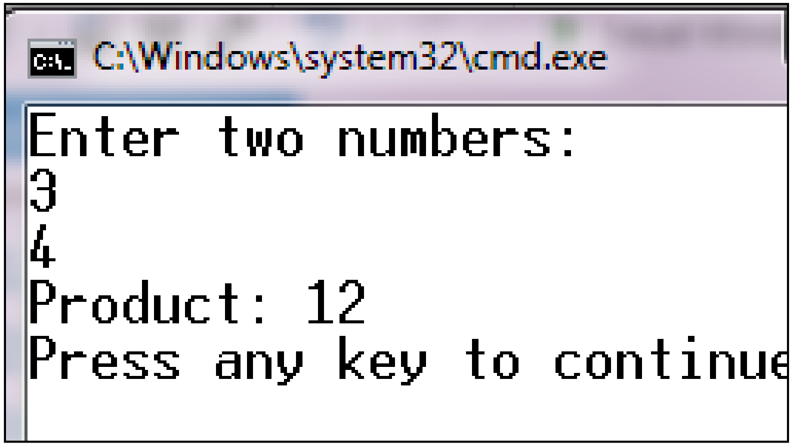
```
int n=4, team1Goals, team2Goals, team1Points=0,team2Points=0;
for (int i=1; i<=n; i++)
{
   cout << "Enter result of match number "<< i <<": \n";
   cin >> team1Goals >> team2Goals;
   if (team1Goals == team2Goals)
   {
      team1Points++;
      team2Points++;
   }
   else if (team1Goals > team2Goals)
      team1Points += 3;
   else
      team2Points += 3;
}
if (team1Points > team2Points)
   cout << "Team 1 wins\n";
else if (team1Points < team2Points)
   cout << "Team 2 wins\n";
else
   cout << "Tie\n";
```



```
Enter result of match number 1:
12
45
Enter result of match number 2:
23
66
Enter result of match number 3:
21
55
Enter result of match number 4:
23
22
Team 2 wins
Press any key to continue . . .
```

Problem: Product Using Sum

- Implement a program to compute the product of two numbers by the sum operation using the for loop.



```
Enter two numbers:
3
4
Product: 12
Press any key to continue
```

- Note:

  6x4 = 6+6+6+6= 24

  3x7= 3+3+3+3+3+3+3= 21

Product Using Sum

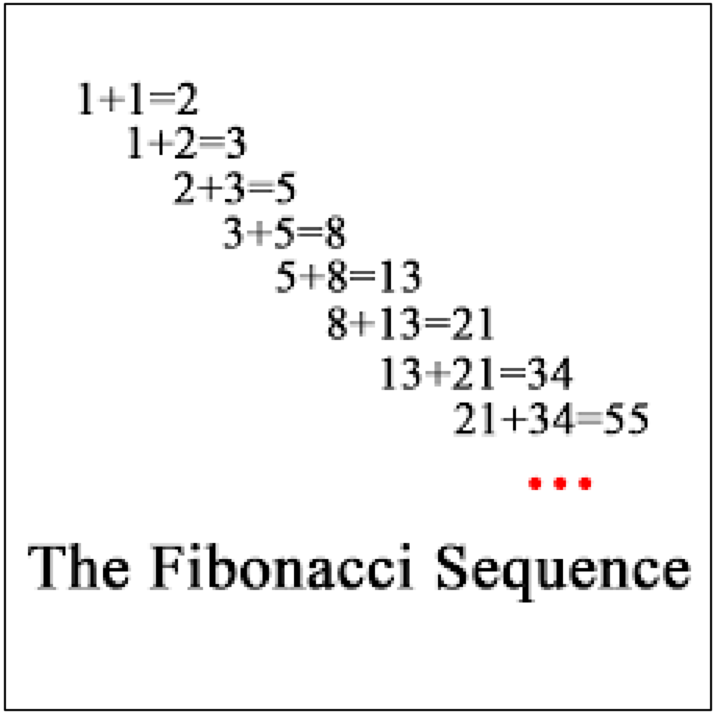
```
#include <iostream>
using namespace std;
int main ()
{
    int num1, num2, result=0;
    cout << "Enter two numbers:\n";
    cin >> num1 >> num2;
    for (int i = 1; i <= num2; i++)
                            result  +=  num1;
    cout << "Product: " << result << endl;
    return 0;
}
```

Problem: Fibonacci Sequence

- Program that displays the first N Fibonacci sequence values after 0 and 1

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, …..

$$F_n = F_{n\text{-}1} + F_{n\text{-}2}$$



1+1=2
1+2=3
2+3=5
3+5=8
5+8=13
8+13=21
13+21=34
21+34=55
• • •

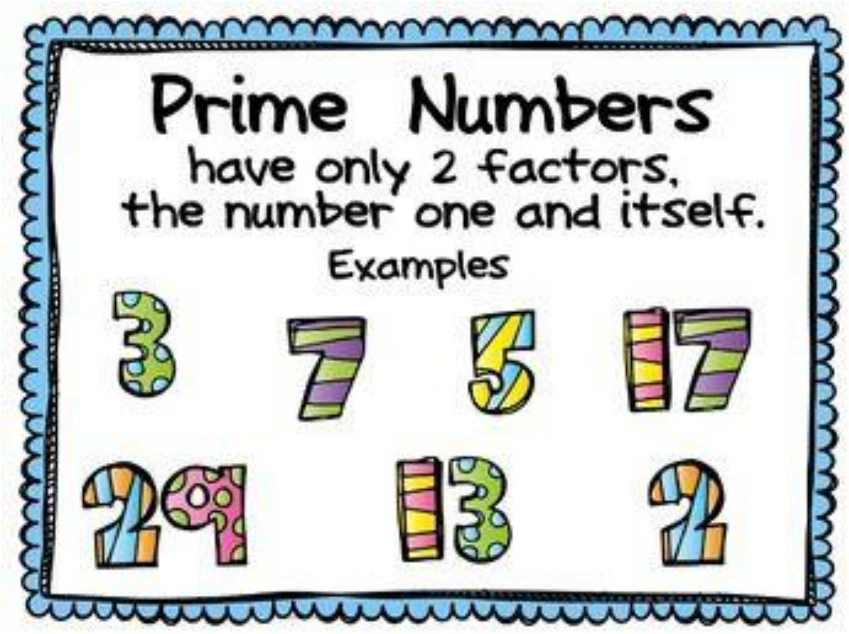
The Fibonacci Sequence

Fibonacci Sequence

```
int N, first = 0, second = 1, next;
cout << "How many numbers: ";
cin >> N;

cout << first  << "  " << second << "  ";

for (int i = 1; i <= N; i++)
{
   next = first + second;
   cout << next << " ";
   first = second;
   second = next;
}
```

Problem: Prime Number

- Program that checks if an input number is prime or not.



Prime Numbers

have only 2 factors, the number one and itself.

Examples

Prime Number

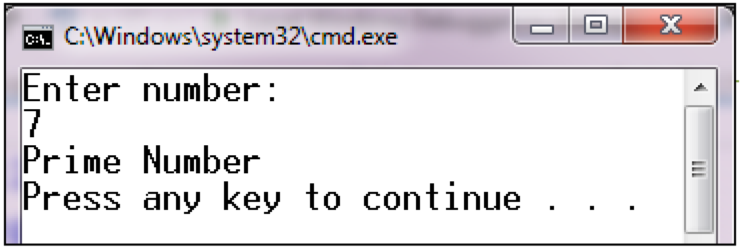
```
int number; // Number to be tested
cout <<"Enter number: \n";
cin >> number;

// Test if number is prime
for (int divisor = 2; divisor < number; divisor++)
{
   if (number % divisor == 0)
  {
     // If true, the number is not prime
     cout<< "Not Prime\n";
     break; // Exit the for loop
  }
}
cout << "Prime\n";
```

Prime Number

```cpp
int number; // Number to be tested
cout <<"Enter number: \n";
cin >> number;
// Test if number is prime
for (int divisor = 2; divisor < number; divisor++)
{
  if (number % divisor == 0)
  {
     // If true, the number is not prime
     cout<< "Not Prime\n";
     break; // Exit the for loop
  }
  else
     cout << "Prime\n";
}
```

Prime Number

```
int number; // A number to be tested
cout <<"Enter number: \n";
cin >> number;

bool isPrime = true; // Assume the current number prime
// Test if number is prime
for (int divisor = 2; divisor < number; divisor++)
{
    if (number % divisor == 0)
   {
      // If true, the number is not prime
      isPrime = false; // Set isPrime to false
      break; // Exit the for loop
   }
 }
if (isPrime)
      cout << "Prime Number\n";
else
      cout << "Not Prime\n";
```



```
Enter number:
7
Prime Number
Press any key to continue . . .
```

Problem: Prime Numbers

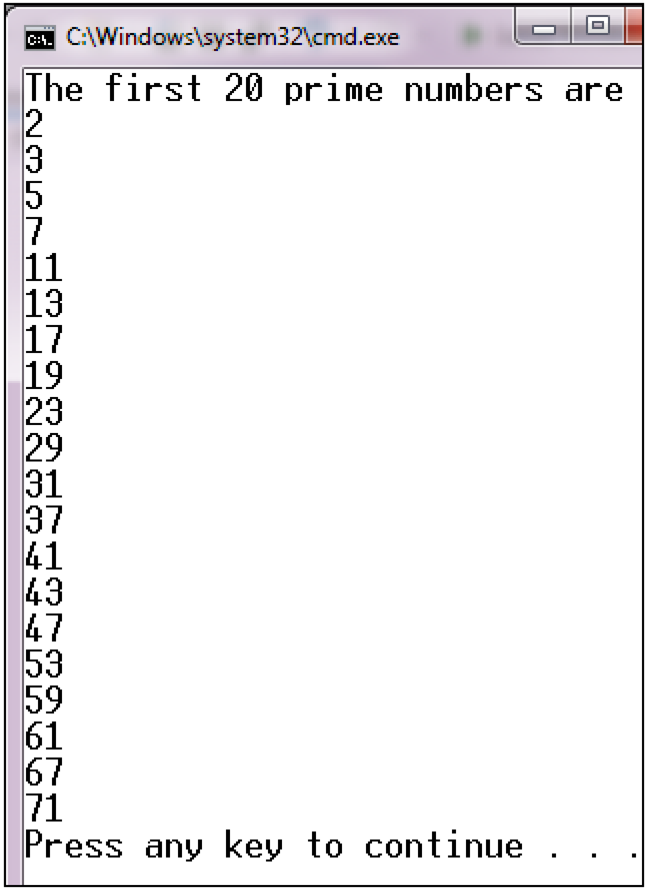
- Write a program that checks 20 numbers and displays whether prime or not.

Prime Numbers

```
int count = 0; // keep track of checked numbers
int number;   // Number to be tested

while (count < 20)
{
   cout << "Enter a number: \n";
   cin >> number;
   bool isPrime = true; // Assume number is prime
   // Test if number is prime
   for (int divisor = 2; divisor < number; divisor++)
   {
     if (number % divisor == 0)
    {
         // If true, the number is not prime
         isPrime = false; // Set isPrime to false
         break; // Exit the for loop
     }
   }
  if (isPrime)
      cout << number << " is a prime number\n";
  else
      cout << number << " is not prime\n";
 count++;
 }
```

Problem: Prime Numbers

- Print the first 20 prime numbers.



```
The first 20 prime numbers are
2
3
5
7
11
13
17
19
23
29
31
37
41
43
47
53
59
61
67
71
Press any key to continue . . .
```

Prime Numbers

```
int count = 0; // Count the number of prime numbers
int number = 2; // First Number to be tested
cout << "The first 20 prime numbers are \n";

while (count < 20)
{
   bool isPrime = true; // Assume number is prime
   // Test if number is prime
   for (int divisor = 2; divisor < number; divisor++)
   {
     if (number % divisor == 0)
    {
         // If true, the number is not prime
         isPrime = false; // Set isPrime to false
         break; // Exit the for loop
     }
   }
   // Display the prime number and increase the count
   if (isPrime)
   {
       count++; // Increase count of prime numbers
       cout << number << endl;
   }
   number++; // get the following number to be tested
}
```

Problem: Least Common Multiple

- Write a program that takes two positive integers as input and computes their least common multiple.
- The *least common multiple (LCM) of two integers is the smallest number* that is evenly divisible by both those numbers.

```
Enter two numbers: 6 8
The LCM is 24
```

x	num1	num2
	6	8
1	6	
2	12	
3	18	
4	24	√

Least Common Multiple

```
int num1, num2, lcm;
int x = 1; // test if number multiple to other number
cout << " Enter two numbers:\n";
cin >> num1 >> num2;

while ( (x*num1) % num2 != 0)
     x++;

lcm = num1 * x;

cout << "The LCM is: " << lcm << endl;
```

Least Common Multiple

x	num1	num2
	12	30
1	12	
2	24	
3	36	
4	48	
5	60	√

x	num1	num2
	30	12
1	30	
2	60	√

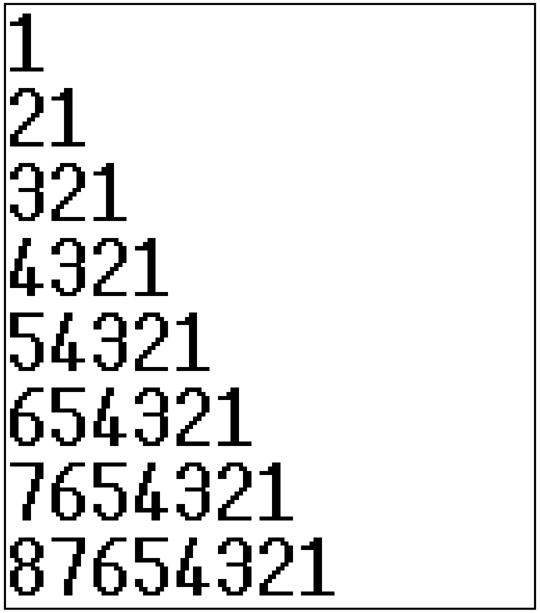
Least Common Multiple

```
int num1, num2, lcm;
int x = 1; // test if number multiple to other number
cout << " Enter two numbers:\n";
cin >> num1 >> num2;
if (num1 < num2)  // swap two numbers
{
      int temp;
      temp = num1; num1 = num2; num2 = temp;
}
while ( (x*num1) % num2 != 0)
      x++;
lcm = num1 * x;
cout << "The LCM is: " << lcm << endl;
```

Problem: Triangle of Numbers

```
for (int lineNum = 1; lineNum <=  8; lineNum++)
{
        cout << lineNum;
  cout << endl;
}
```

✗



```
1
21
321
4321
54321
654321
7654321
87654321
```

Nested Loops

- **Loops may be nested**, with one loop sitting in the body of another.
- The inner loop will be executed in full for every execution of the outer loop.

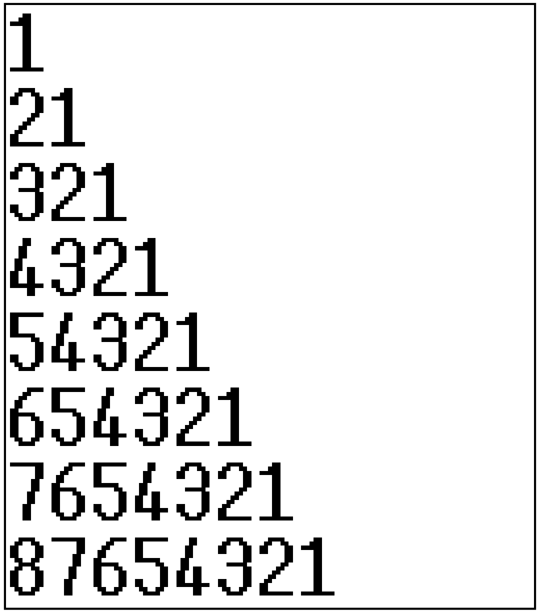
Ex:

```
    for(int i = 0; i < 10; i++)
    {
        for(int j = 1; j < 3; j++)
        {
            ............
            ............
        }
    }
```

i	j
0	1
	2
1	1
	2
2	1
	2
..	..
9	1
	2

Problem: Triangle of Numbers

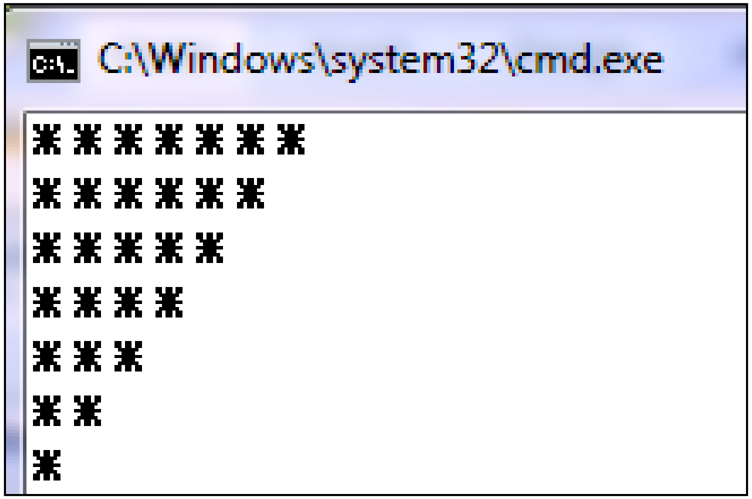
```
for (int lineNum = 1; lineNum <=  8; lineNum++)
{
   for (int  number =  lineNum; number >=  1; number--)
          cout << number;
   cout << endl;
}
```



```
1
21
321
4321
54321
654321
7654321
87654321
```

Problem: Triangle of Stars

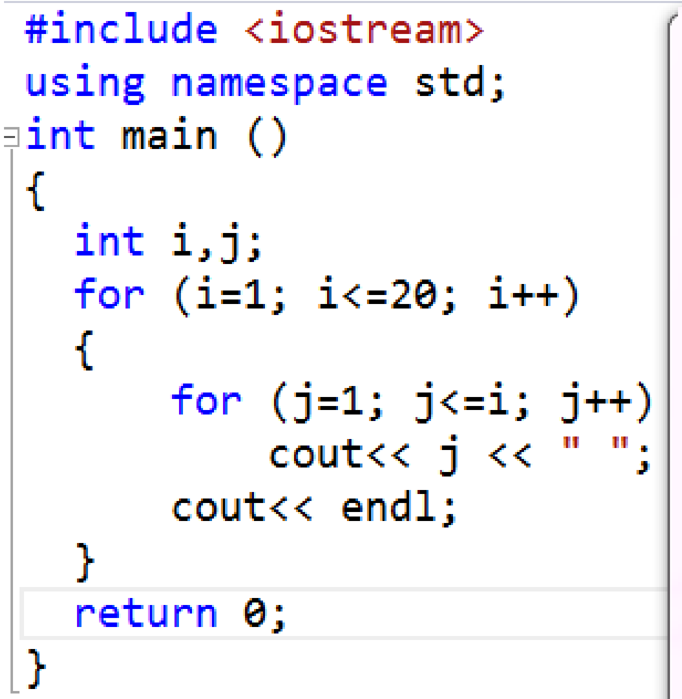
```
for (int numLines = 7; numLines >= 1; numLines--)
{
  for (int stars=1; stars <= numLines; stars++)
      cout << '*';
  cout << endl;
}
```



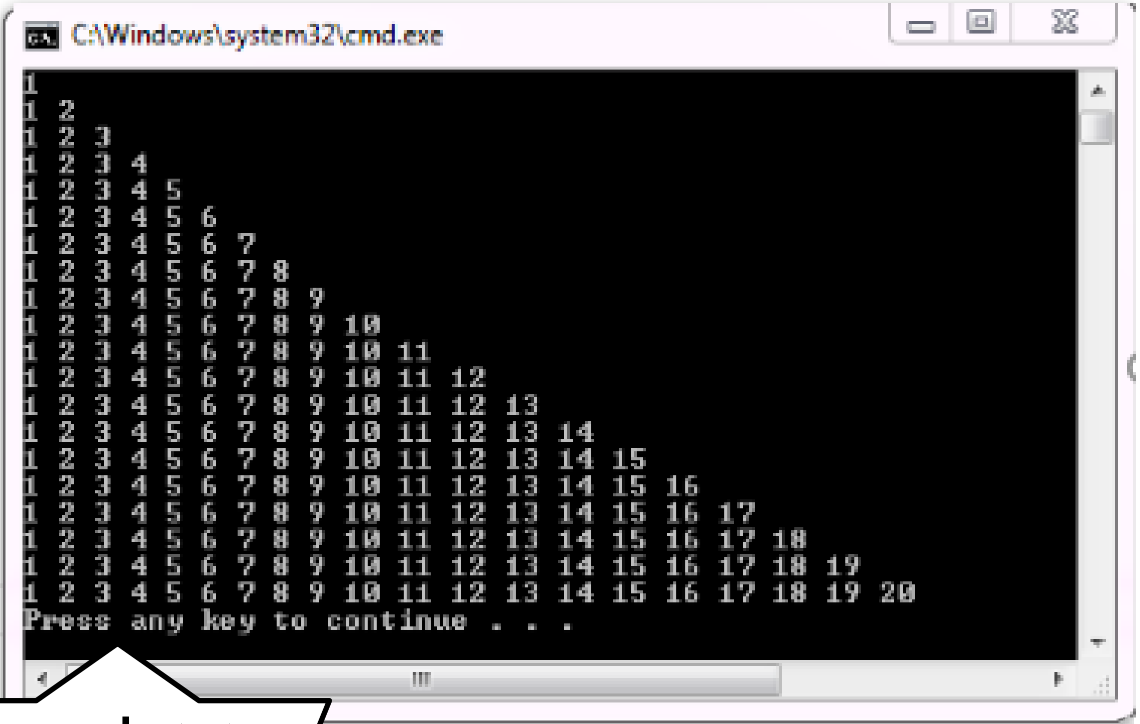
```
*******
******
*****
****
***
**
*
```

Triangle of Numbers

- Write a program that produces this output.



```
#include <iostream>
using namespace std;
int main ()
{
   int i,j;
   for (i=1; i<=20; i++)
   {
        for (j=1; j<=i; j++)
              cout<< j << " ";
        cout<< endl;
   }
   return 0;
}
```



```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
1 2 3 4 5 6
1 2 3 4 5 6 7
1 2 3 4 5 6 7 8
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9 10
1 2 3 4 5 6 7 8 9 10 11
1 2 3 4 5 6 7 8 9 10 11 12
1 2 3 4 5 6 7 8 9 10 11 12 13
1 2 3 4 5 6 7 8 9 10 11 12 13 14
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
Press any key to continue . . .
```

cout<<
i;

Nesting: Characters

- Write C++ program that displays the following:

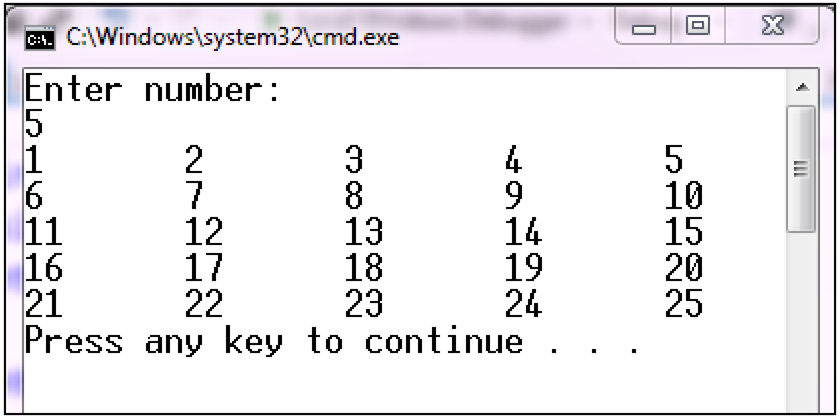
```
ABCDEF
ABCDE
ABCD
ABC
AB
A
```

Solution

```
for (int i = 6; i >= 1; i--)
{
  char x = 'A';
  for (int j = 1; j <= i ; j++)
       cout << x++ ;
  cout << endl;
}
```

Problem: Numbers Grid

- Write a program that asks the user to enter a number N and then prints on the screen the numbers 1 through N^2 arranged in a NxN grid.
- An example is shown below, where the user entered 5.



```
Enter number:
5
1       2       3       4       5
6       7       8       9       10
11      12      13      14      15
16      17      18      19      20
21      22      23      24      25
Press any key to continue . . .
```

Numbers Grid

```
int num;
cout << "Enter number:\n";
cin >> num;

int n = 1;
for (int x = 1 ; x <= num ; x++)
{
     for (int y = 1 ; y <= num ; y++)
           cout << n++ << '\t';
     cout << endl;
}
```

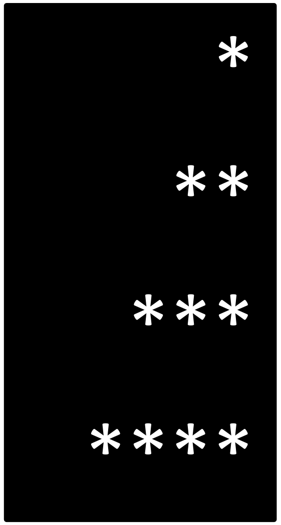
Numbers Grid

```
int num;
cout << "Enter number:\n";
cin >> num;

For (int x = 1 ; x <= num*num ; x++)
{
     cout << x << '\t';
     if (x % num == 0)
           cout << endl;
}
```

Nesting: Triangle of Stars

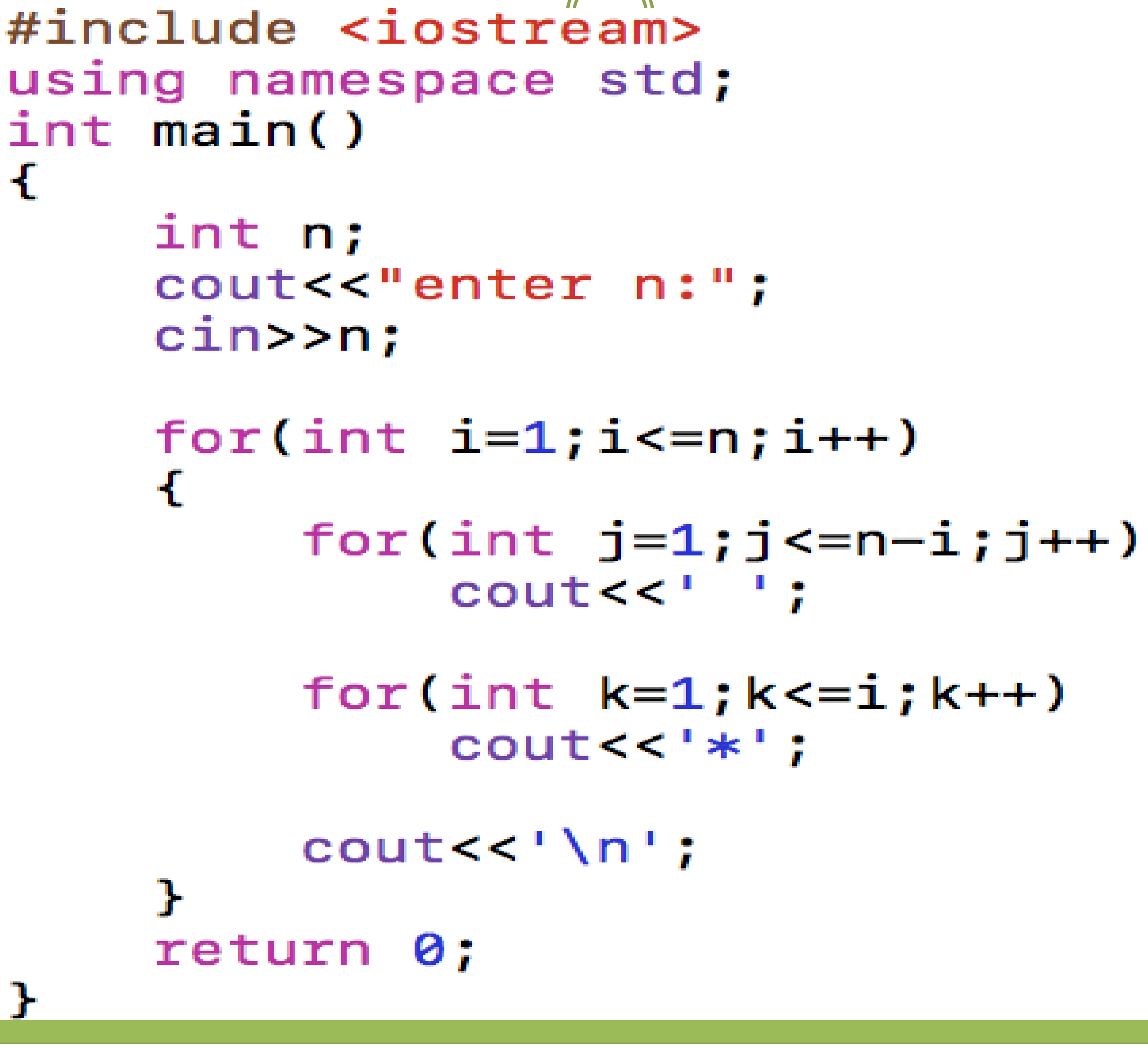
- Write C++ program that draws a triangle from ‘*’ with the following pattern.
- Note: the triangle size is determined by the user.

- Example:



```
   *

  **

 ***

****
```

Solution



```
#include <iostream>
using namespace std;
int main()
{
    int n;
    cout<<"enter n:";
    cin>>n;

    for(int i=1;i<=n;i++)
    {
        for(int j=1;j<=n-i;j++)
            cout<<' ';

        for(int k=1;k<=i;k++)
            cout<<'*';

        cout<<'\n';
    }
    return 0;
}
```

Clean Code

- *“Clean code is code that is easy to understand and easy to change”*
- "Code is like humor. When you *have* to explain it, it’s bad"
- There are some ideas that are considered best practice and what constitutes as clean code within the industry and community, but there is no definitive distinction.
  - It works correctly
  - It is easy to understand the execution flow of the entire application
  - It is easy to understand what is the purpose of each expression and variable

Clean Code, why?

- If you write clean code, then you are helping your future self and your co-workers.
- You are reducing the cost of maintenance of the application you are writing.
- You are making it easier to estimate the time needed for new features.
- You are making it easier to fix bugs.
- You are making it more enjoyable to work on the code for many years to come.
- Essentially you are making the life easier for everyone involved in the project.

Best Practices

- Use comments (when needed)
- Assign intuitive names to variables and functions
- Don't make your code unnecessarily complex
- Apply rules uniformly

Programming Style Guidelines

- Include statements must be located at the top of file only.
- Include statements should be sorted and grouped.
- Variable names must be in mixed case starting with lower case.
- Plural form should be used on names representing a collection of objects.
- The prefix *n* should be used for variables representing a number of objects. E.g. *nLines*
- Abbreviations in names should be avoided.

Source: http://geosoft.no/development/cppstyle.html

Programming Style Guidelines

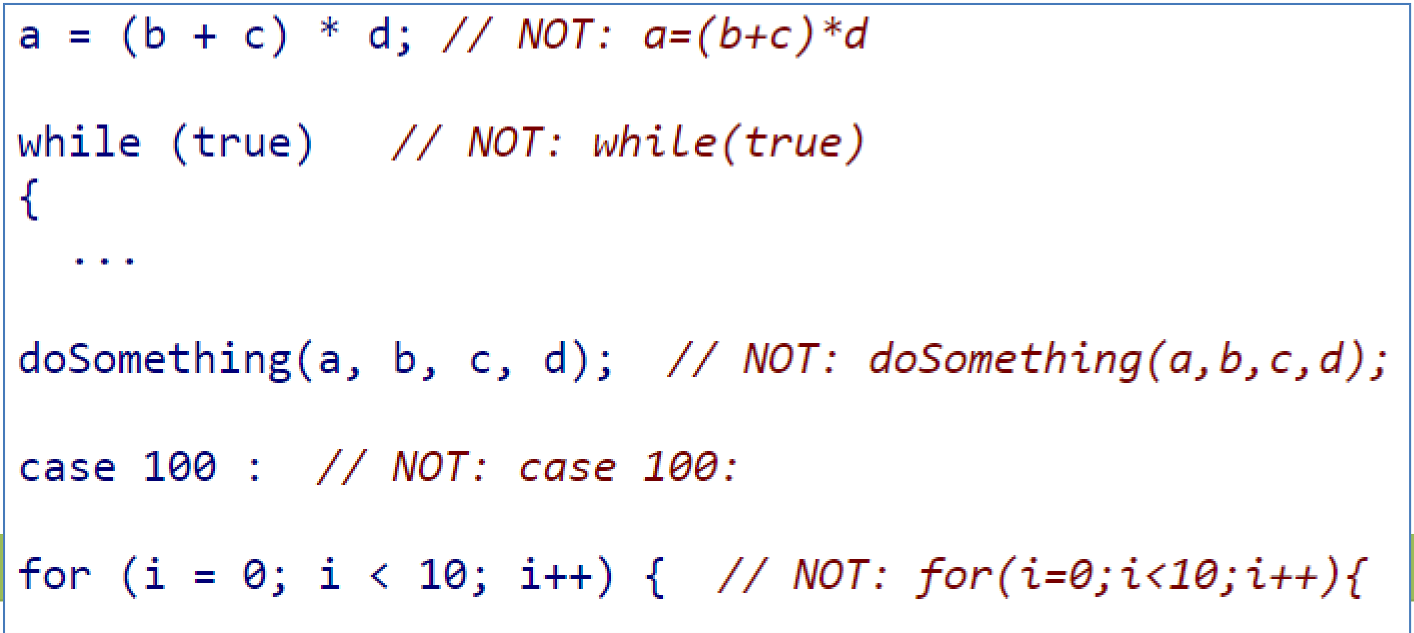
- The prefix *is* should be used for Boolean variables and methods. E.g. *isPrime*, *isEven*.
- Negated Boolean variable names must be avoided.
- The incompleteness of split lines must be made obvious.
  - Break after a comma.
  - Break after an operator.
  - Align the new line with the beginning of the expression on the previous line.
- Variables should be initialized where they are declared.
- Variables should be declared in the smallest scope possible.

Programming Style Guidelines

- Use indentation.
- Single statement if-else, for or while statements can be written without brackets.
- The supposed case should be put in the if-part and the exception in the else-part of an if statement.
- The condition should be put on a separate line.

Programming Style Guidelines

- White Space:
  - Conventional operators should be surrounded by a space character.
  - C++ reserved words should be followed by a white space.
  - Commas should be followed by a white space.
  - Colons should be surrounded by white space.
  - Semicolons in for statements should be followed by a space character.



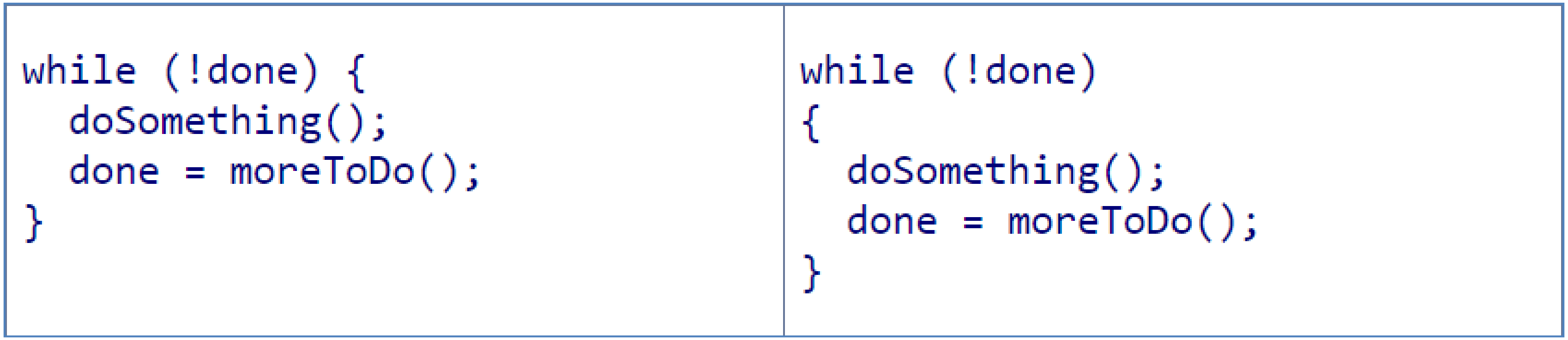
```
a = (b + c) * d; // NOT: a=(b+c)*d

while (true)   // NOT: while(true)
{
  ...

doSomething(a, b, c, d);  // NOT: doSomething(a,b,c,d);

case 100 :  // NOT: case 100:

for (i = 0; i < 10; i++) {  // NOT: for(i=0;i<10;i++){
```

Programming Style Guidelines

- Floating point constants should always be written with decimal point and at least one decimal. E.g. double grade = 50.0;
- Loop variables should be initialized immediately before the loop.
- The form *while (true)* should be used for infinite loops.
- Iterator variables should be called *i*, *j*, *k*, etc.

Programming Style Guidelines

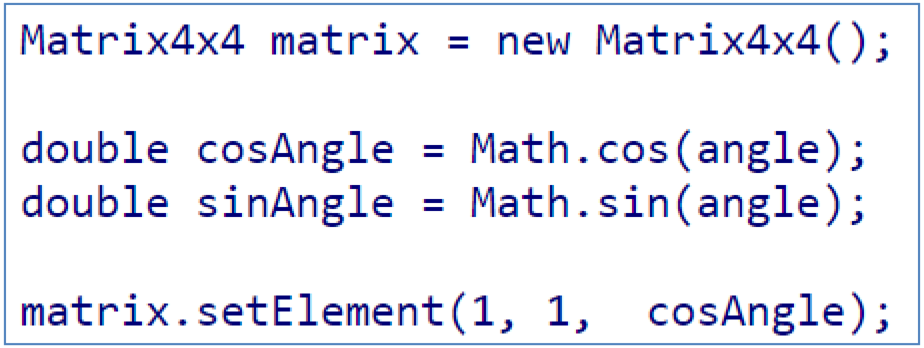
- Block layout should follow one of the following. Functions and class blocks should follow ex2.



```
while (!done) {
  doSomething();
  done = moreToDo();
}
```

```
while (!done)
{
  doSomething();
  done = moreToDo();
}
```

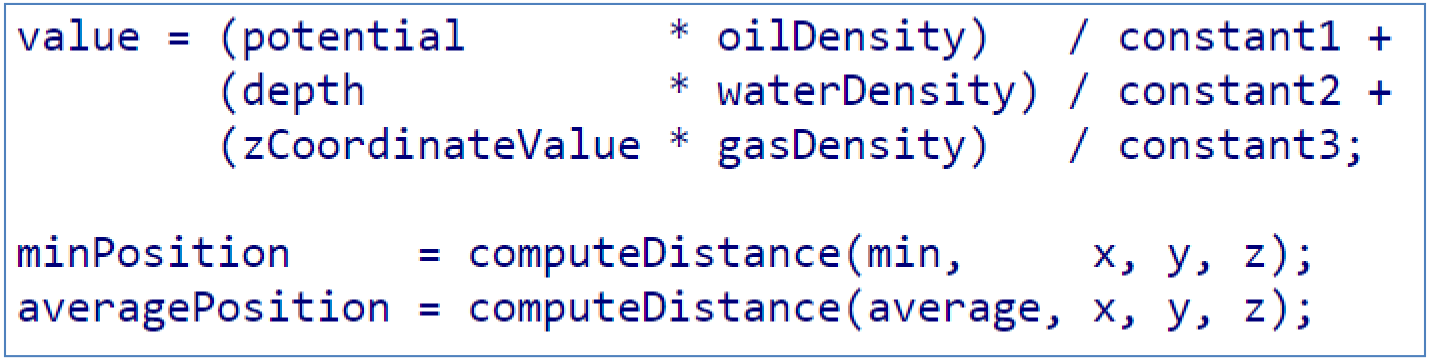
- Logical units within a block should be separated by one blank line.



```
Matrix4x4 matrix = new Matrix4x4();

double cosAngle = Math.cos(angle);
double sinAngle = Math.sin(angle);

matrix.setElement(1, 1,  cosAngle);
```

Programming Style Guidelines

- Use alignment wherever it enhances readability.

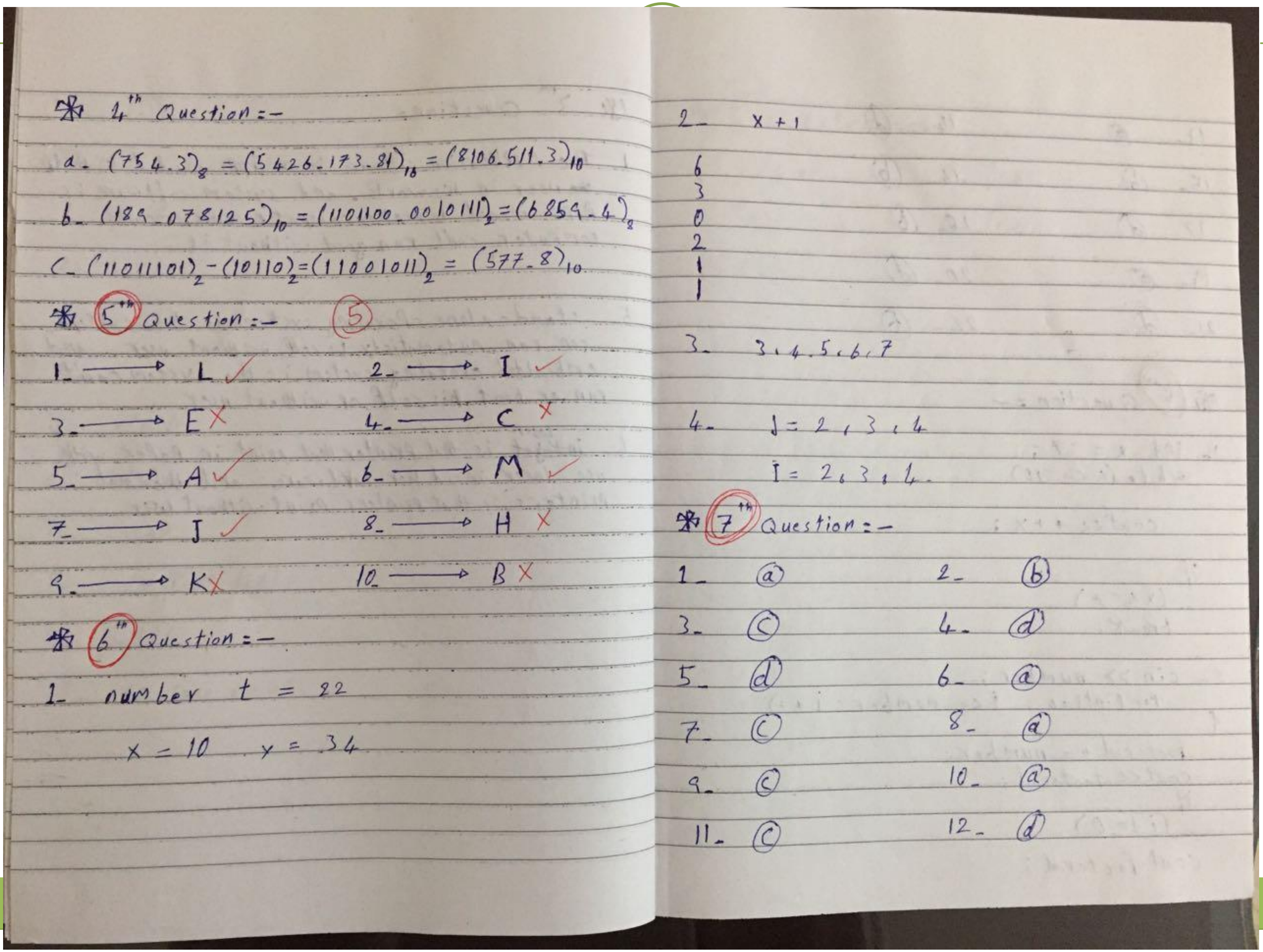


```
value = (potential         * oilDensity)   / constant1 +
        (depth             * waterDensity) / constant2 +
        (zCoordinateValue  * gasDensity)   / constant3;

minPosition     = computeDistance(min,     x, y, z);
averagePosition = computeDistance(average, x, y, z);
```

General Guidelines

- Arrive on time.
- Bring Faculty ID or any other ID.
- No mobiles.
- Bring your pen, pencil, eraser, etc.
- Do not use the exam paper as a draft. It is not allowed to write on it.
- You can use a page in answer paper as a draft.
- Write in answer paper from left to right.
- Answer only one question on each page.
- Write question number clearly on top of page.

Answer only one question on each page!!



✻ 4th Question :-

a- $(754.3)_8 = (5426.173.81)_{16} = (8106.511.3)_{10}$

b- $(189.078125)_{10} = (1101100.0010111)_2 = (6859.4)_8$

c- $(11011101)_2 - (10110)_2 = (11001011)_2 = (577.8)_{10}$

✻ 5th Question :- 5

1- → L ✓ 2- → I ✓

3- → E ✗ 4- → C ✗

5- → A ✓ 6- → M ✓

7- → J ✓ 8- → H ✗

9- → K ✗ 10- → B ✗

✻ 6th Question :-

1- number t = 22

x = 10 y = 34

2- x + 1

6
3
0
2
1
1

3- 3, 4, 5, 6, 7

4- J = 2, 3, 4

I = 2, 3, 4.

✻ 7th Question :-

1- ⓐ 2- ⓑ

3- ⓒ 4- ⓓ

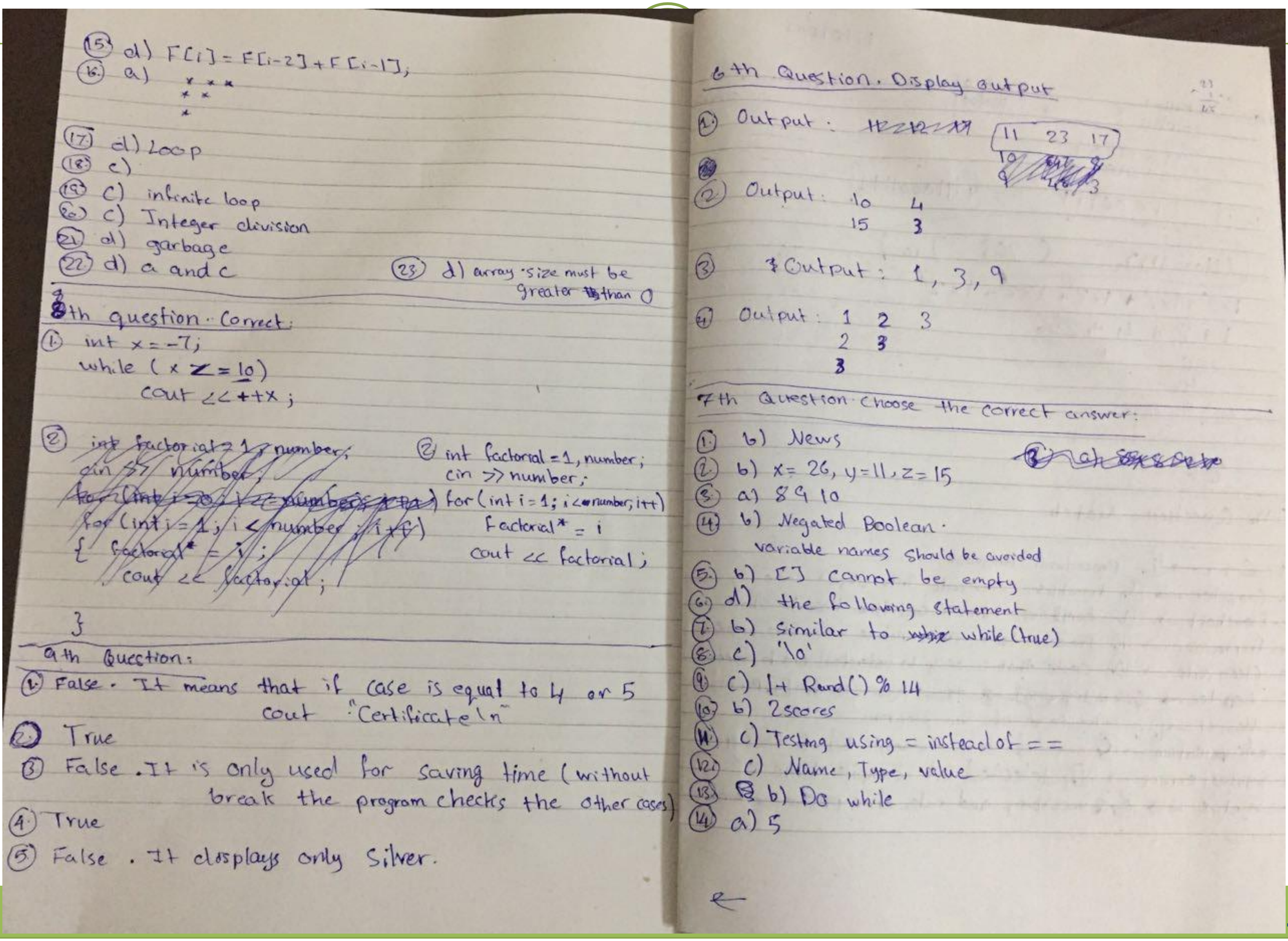
5- ⓓ 6- ⓐ

7- ⓒ 8- ⓐ

9- ⓒ 10- ⓐ

11- ⓒ 12- ⓓ

!!!



(15) d) F[i] = F[i-2] + F[i-1];

(16) a)

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

(17) d) Loop

(18) c)

(19) c) infinite loop

(20) c) Integer division

(21) d) garbage

(22) d) a and c

(23) d) array size must be greater than 0

8th question. Correct:

(1)

```
int x = -7;
while ( x <= 10)
    cout << ++x ;
```

(2)

~~int factorial = 1, number;~~
~~cin >> number;~~
~~for (int i = 0; i <= number; i++)~~
~~for (int i = 1; i < number ; i++)~~
~~{ factorial* = i ;~~
~~cout << factorial ;~~
}

(2)

```
int factorial = 1, number;
cin >> number;
for (int i = 1; i <= number; i++)
    Factorial* = i
    cout << factorial;
```

9th Question:

(1) False. It means that if case is equal to 4 or 5 cout "Certificate\n"

(2) True

(3) False. It is only used for saving time (without break the program checks the other cases)

(4) True

(5) False. It displays only Silver.

6th Question. Display output

(1) Output: ~~11 23 17~~ 11 23 17

(2) Output:

```
10   4
15   3
```

(3) Output: 1, 3, 9

(4) Output:

```
1  2  3
2  3
3
```

7th Question Choose the correct answer:

(1) b) News

(2) b) x = 26, y = 11, z = 15

(3) a) 8 9 10

(4) b) Negated Boolean. variable names should be avoided

(5) b) [ ] cannot be empty

(6) d) the following statement

(7) b) similar to ~~while~~ while (true)

(8) c) '\0'

(9) c) 1+ Rand() % 14

(10) b) 2scores

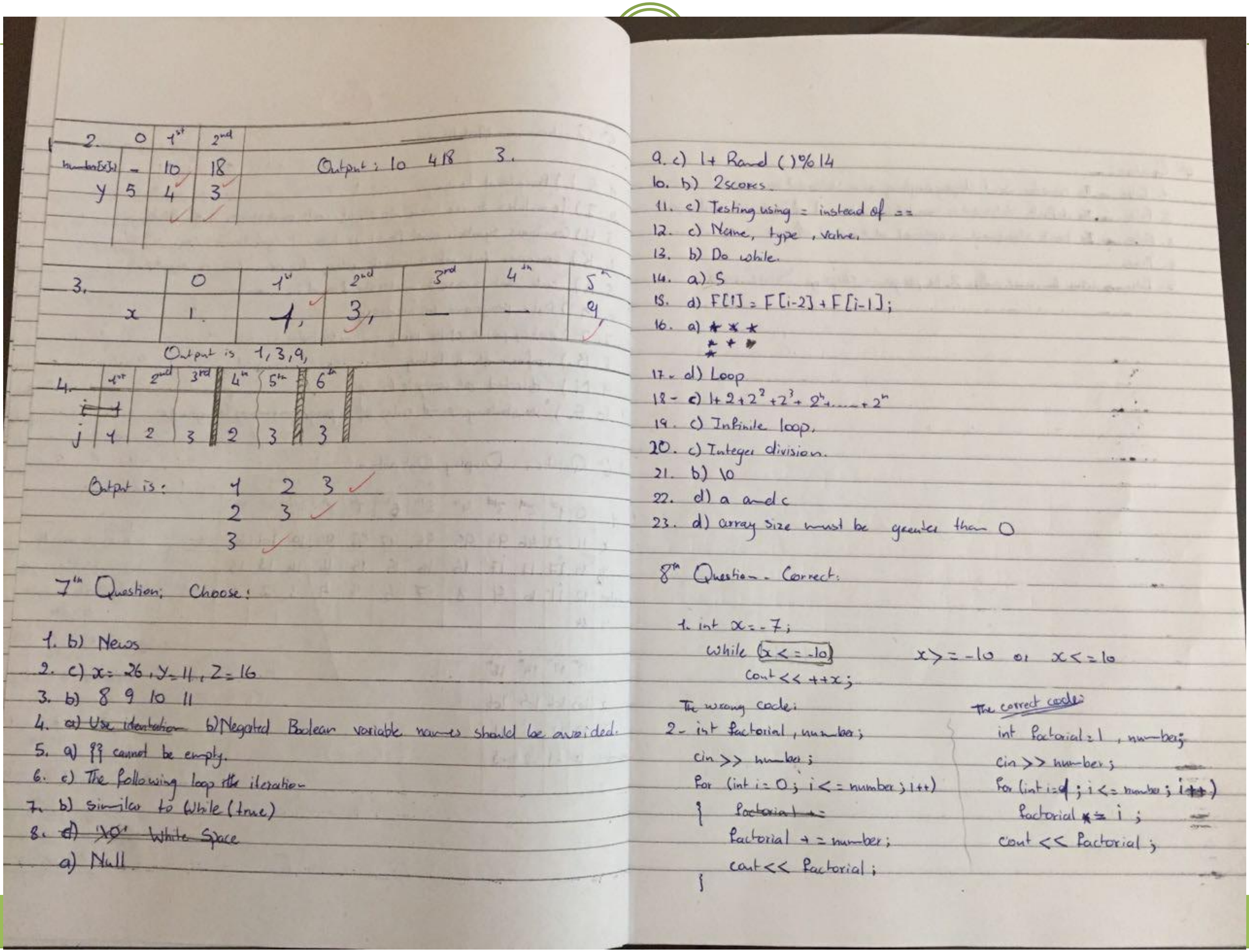
(11) c) Testing using = instead of ==

(12) c) Name, Type, value

(13) b) Do while

(14) a) 5

Blank Lines



2.	0	1st	2nd
number[x][x]	-	10 ✓	18 ✓
y	5	4 ✓	3 ✓

Output : 10 418 3.

3.	0	1st	2nd	3rd	4th	5th
x	1.	1, ✓	3, ✓	—	—	9, ✓

Output is 1, 3, 9,

4.	1st	2nd	3rd	4th	5th	6th
i	~~1~~					
j	1	2	3	2	3	3

```
Output is :   1  2  3 ✓
              2  3 ✓
              3 ✓
```

7th Question; Choose:

1. b) News
2. c) x = 26 , y = 11 , Z = 16
3. b) 8 9 10 11
4. ~~a) Use identation~~ b) Negated Boolean variable names should be avoided.
5. a) {} cannot be empty.
6. c) The following loop the iteration
7. b) similar to While (true)
8. ~~d) '\0' White Space~~
   a) Null
9. c) 1+ Rand ()%14
10. b) 2scores.
11. c) Testing using = instead of ==
12. c) Name, type , value,
13. b) Do while.
14. a) 5
15. d) F[i] = F[i-2] + F[i-1];
16. a)

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

17. d) Loop
18. c) $1+2+2^2+2^3+2^4+....+2^n$
19. c) Infinite loop.
20. c) Integer division.
21. b) 10
22. d) a and c
23. d) array size must be greater than 0

8th Question - Correct:

```
1. int x = -7;
     while (x <= -10)                      x >= -10   or   x <= 10
         cout << ++x;
```

The wrong code:

```
2- int factorial , number;
   cin >> number;
   For (int i = 0; i <= number; i++)
   {  ~~Factorial *=~~
      Factorial += number;
      cout << Factorial;
   }
```

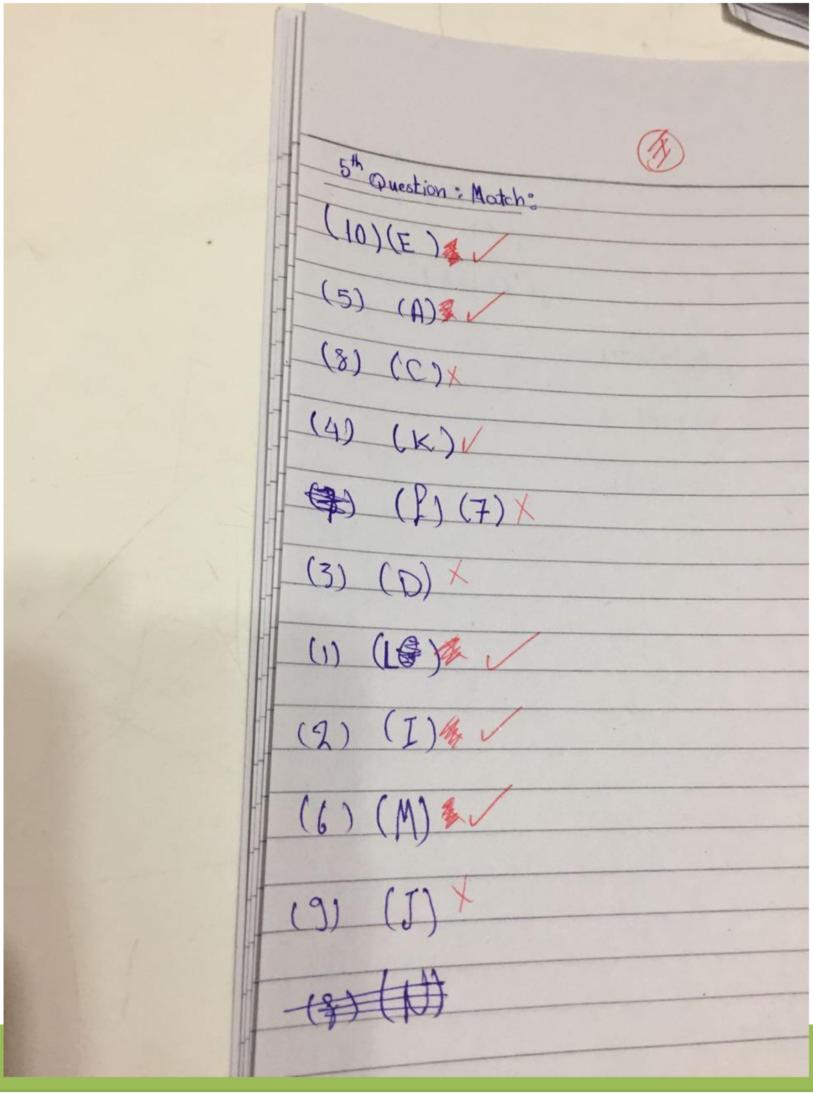
The correct code:

```
int Factorial = 1 , number;
cin >> number;
For (int i = 1 ; i <= number ; i++)
   Factorial *= i ;
cout << Factorial ;
```

Guidelines for Our Final Exam

- Final exam covers all lectures and C++ labs.
- No calculators.
- Do not provide two answers for one question.
- Write question number and answer number/letter clearly i.e. A/B/C/D…
- Answer different parts in each question sequentially, especially MCQ, complete and true/false.

Answer questions in order



5th Question : Match:

(10)(E) ✓

(5) (A) ✓

(8) (C) ✗

(4) (K) ✓

(F) (7) ✗

(3) (D) ✗

(1) (L) ✓

(2) (I) ✓

(6) (M) ✓

(9) (J) ✗

~~(8) (N)~~

Exam Questions

- Complete
- MCQ
- Compare
- Convert
- True/False
- Match
- Draw flowchart
- Correct and justify
- Trace/Display output
- Write a C++ program

Complete

1. C-preprocessor

1. ……………………………….is a text substitution tool to do required pre-processing before actual compilation.
2. Assembly is considered ……………………… generation language.
3. ……………………………… can only process the input from the keyboard once the RETURN key has been pressed.
4. ………………… errors won’t be displayed on the screen, they are caused by incorrect business requirement understanding that will lead to display wrong results.
5. The ……………………… operator can be used as a prefix or suffix.
6. ………………………… makes information hiding (encapsulation) possible.

Match

1. C. Prolog

A

1. Natural language
2. Mnemonics
3. Algorithm
4. Desk checking
5. Connectors
6. Component reusability
7. Interpreter
8. Machine language

B

A. Assembly language code

B. First generation

C. Prolog

D. Creates an object code

E. Programmers do not have to wait for the entire program to be recompiled each time they make a change.

F. OOP

G. Steps of solution

H. Design tool

I. Links breaks in flowcharts

Correct and Justify

```
int x= 0;
while (x > = 0);
{
        cout<<  Enter a number;
        cin  >> x;
}
```

Rewrite the code after correction along with justification

```
char answer;
do
{
        cout<<"Do you want to continue?";
        cin   >> answer;
} while (answer = 'Y' && answer = 'y');
```

Display Output

```
int  y=3;
int  numbers[8]={3,6,3,4,9,5,3,6};
for (int x=2; x>=2; x+=2)
{
            cout<< x << '\t' << y << endl;
            x = numbers[x-1] - numbers[x];
            y = x*2;
}
```

x	y
2	3
5	6
6	8
4	4

Display **only** the output (cout) that will be displayed on screen.

Do not display **all** values of variables during desk checking/tracing.

Trace

- Display **all** values of **all** variables during desk checking/tracing.

zzz..
*!
eat. sleep. revise. And repeat.