

النوت 20 صفحة سحب كمبيوتر:
ملونة دينار كويتي واحد
أبيض وأسود: ربع دينار فقط

C++ Programming

Week 4

Returning-Value Functions

الشرح والأفكار الرئيسية

نوت السي ++

يحتوي على شرح المواضيع
وأمثلة للأفكار الرئيسية من
هوموركات وامتحانات سابقة.

نوت السي ++

تتكون النوت من عشرة
أجزاء بمعدل جزء كل أسبوع.

نوت السي ++

يتوفر على الموقع ملفات الشرح
والأفكار الرئيسية، وملفات
لتمارين وأمثلة إضافية.

لماذا لا تقتنى الأحدث؟

النوت يتم تنقيحها وتحديثها نهاية
كل أسبوع، راجع eng-hs.net
للتأكد من شرائك الإصدار الأحدث.

المنافسة الواقعية تكون بين ما
تقوم بعمله وما أنت قادر على
عمله، قارن نفسك مع نفسك
وليس مع أي شخص آخر.

Functions (*Introduction*)

Write a program that reads 2 integers and prints the sum of integers in between inclusively (without using functions).

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

```
int main ( )
{
    int x, y, i, min, max, sum = 0;
```

```
    cout << "enter 2 integers: ";
```

```
    cin >> x >> y;
```

تم كتابة هذا البرنامج بالطريقة المعتادة بدون استخدام دوال.

```
    if (x < y)
```

```
        { min = x; max = y; }
```

```
    else
```

```
        { min = y; max = x; }
```

```
    for (i = min ; i <= max ; i++)
```

```
        sum += i;
```

```
    cout << "sum is: " << sum << endl;
```

```
    return 0;
```

```
}
```

enter 2 integers: **10 3**
sum is: **52**

enter 2 integers: **3 10**
sum is: **52**

i	sum = 0
3	3
4	7
5	12
6	18
7	25
8	33
9	42
10	52

مطلوب كتابة نفس البرنامج بالصفحة التالية باستخدام (Functions).

كون البعض ناجحون
يثبت أن الآخرين يمكنهم
أن يكونوا ناجحين أيضاً.

Functions (Introduction)

Write a program that reads 2 integers and calls a function that gets these two integers and returns the sum of integers in between.
The main function should print the result.

```
#include <iostream>
using namespace std;
int fun (int, int);
int main ()
{
```

enter 2 integers: 10 3
sum is: 52

enter 2 integers: 3 10
sum is: 52

```
int x, y, sum;
```

```
cout << "enter 2 integers: ";
```

```
cin >> x >> y;
```

```
sum = fun (x, y);
```

```
cout << "sum is: " << sum << endl;
```

```
return 0;
```

```
}
```

اسم اختياري

function name

استدعاء الدالة

function call

نفس البرنامج السابق
مع استخدام دالة.

معاملات الدالة

function parameters

function return type

مقدمة الدالة

function header

```
int fun (int a, int b)
{
```

```
int i, min, max, s = 0;
```

```
if (a < b)
```

```
{ min = a; max = b; }
```

```
else
```

```
{ min = b; max = a; }
```

```
for (i = min; i <= max; i++)
```

```
s += i;
```

```
return s;
```

```
}
```

function body

الجزء المنقول من main

يمكنك الإبقاء على نفس الأسماء (x, y) أو
تغييرها (s) تقابل (sum) بالـ (main)

function return value

القيمة العائدة من الدالة

لا يتعب المرء إلا إذا
توقف وكان عنده ما
يكفي من الوقت لذلك.

Functions (Power)

Write a program that reads two positive integers, and call a function that returns the power of the first integer raised to the second integer.

Note: you are not allowed to use <cmath> library functions.

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

```
int power (int, int);
```

```
int main ( )
```

```
{
```

```
    int x, y, p;
```

```
    cout << "Enter two positive integers: ";
```

```
    cin >> x >> y;
```

```
    p = power (x , y);
```

```
    cout << "The power is: " << p << endl;
```

```
    return 0;
```

```
}
```

```
int power (int x, int y)
```

```
{
```

```
    int i, p = 1;    ← p تبدأ من (1) وليس (0) لأنها ضرب
```

```
    for (i = 1; i <= y; i++)
```

```
        p *= x;
```

```
    return p;
```

```
}
```

Enter two positive integers: **3 4**
The power is: **81**

$$x^y = x . x . x \dots (y \text{ times})$$

$$3^4 = 3 . 3 . 3 . 3 = 81$$

من فوائد الدوال:

- تقسيم العمل على المبرمجين.
- إعادة توظيف الدوال ببرامج أخرى.
- سهولة قراءة وفهم البرامج.

اذهب لحلمك الآن،
فالمستقبل غير
مضمون لأحد.

Functions (Middle)

Write a program that reads three integers and calls a function to find their middle. The main function will print the result.

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

int middle (int, int, int);
```

```
int main ( )
```

```
{
    int x, y, z, mid;
    cout << "Enter 3 integers: ";
    cin >> x >> y >> z;
    mid = middle (x, y, z);
    cout << "Middle number is: " << mid << endl;
    return 0;
}
```

Enter 3 integers: **7 8 3**
Middle number is: **7**

Enter 3 integers: **6 16 6**
Middle number is: **6**

في حال تنفيذ أمر return في أي دالة سيتم الرجوع فوراً إلى main وإهمال باقي أوامر الدالة.

```
int middle (int x , int y, int z)
```

```
{
    if (x >= y && x <= z || x >= z && x <= y)
        return x ;
    if (y >= x && y <= z || y >= z && y <= x)
        return y ;
```

```
        return z; لا نحتاج لكتابة الشرط
}
```

في حال عدم تحقق الشرط الأول وعدم تحقق الشرط الثاني سيتم تنفيذ آخر return.

البحيرات سهلة الوصول إليها
ينتهي الصيد فيها سريعاً، على
عكس تلك صعبة الوصول.

Functions (*Even or odd?*)

Write a program that reads an integer and calls a function that indicates if the integer is even or odd, the main function will print the result.

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

Enter an integer: 25
25 is odd.

```
bool Is_Odd (int);
```

```
int main ( )
{
    int x;
    bool y;

    cout << "Enter an integer: ";
    cin >> x;

    y = Is_Odd (x);

    if (y == true)
        cout << x << " is odd." << endl;
    else
        cout << x << " is even." << endl;

    return 0;
}
```

الدوال التي تعيد نعم أو لا
يفضل أن تكون من نوع bool.
الـ bool يأخذ قيمة true أو false فقط.

```
bool Is_Odd (int x)
{
    if (x % 2 != 0)
        return true;

    return false;
}
```

يكون الرقم (odd) إذا تم قسمته على
(2) وكان الباقي لا يساوي صفراً.

البعض ينجح لذكائه،
والبعض الآخر ينجح
لغباء الآخرين.

Functions (*Factorial*)

Write a program that reads a positive integer and calls a function that calculates its factorial. The main function will print the result.

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

```
int fact (int x)
{
    int i, f = 1;
    for (i = 1; i <= x ; i++)
        f *= i;

    return f;
}
```

Enter an integer: -3
Invalid number!

Enter an integer: 4
Factorial is: 24

تذكر أن متغيرات حاصل الضرب تبدأ بـ (1) وليس بصفر وإلا كان الناتج النهائي صفراً.

```
int main ()
{
    int x;

    cout << "Enter an integer: ";

    cin >> x;

    if (x < 0)
        cout << "Invalid number!\n";
    else
        cout << "Factorial is: " << fact (x) << endl;

    return 0;
}
```

في حال كتابة الدالة قبل الـ main
لا داعي من كتابة prototype.

الخوف من الفشل أهم
سبب لإحجام الكثير
عن تحقيق النجاح.

Functions (*Reverse-Sign*)

Write a program that reads a number and calls a function that reverses the sign of the number received according to the following sample outputs. The main function will print the result.

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

```
double reverse (double x)
{
    return x * -1;
}
```

في حال رقم موجب أو سالب
فإن تغيير إشارته في الحالتين
يكون بضربه في (-1).

Enter a number: 0
No reverse!

Enter a number: 7.1
Reverse is: -7.1

Enter a number: -3.9
Reverse is: 3.9

```
int main ()
{
    double x;
    cout << "Enter a number: ";
    cin >> x;
    if (x == 0)
        cout << "No reverse!" << endl;
    else
        cout << "Reverse is: " << reverse (x) << endl;
    return 0;
}
```

لا تتعجب من كون الدالة قليلة
الكود لأن الهدف منها الآن
هو تعلم كيفية التعامل معها.

العالم بأسره يتنحى جانباً
ليفسح الطريق للإنسان الذي
يعرف تماماً إلى أين يتوجه.

Functions (Perfect)

Write a program that reads a positive integer and calls a function that checks if it is perfect or not. The main function will print the result.

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

```
bool perfect (int);
```

```
int main ( )
```

```
{
```

```
    int x;
```

```
    bool y;
```

```
    cout << "Enter a positive integer: ";
```

```
    cin >> x;
```

```
    y = perfect (x);
```

```
    if (y == true)
```

```
        cout << x << " is perfect." << endl;
```

```
    else
```

```
        cout << x << " is not perfect." << endl;
```

```
    return 0;
```

```
}
```

Enter a positive integer: **28**

28 is perfect.

Enter a positive integer: **16**

16 is not perfect.

يكون الرقم perfect إذا كانت قيمته تساوي مجموع عوامله.

6	16	24
1	1	1
2	2	2
3	4	4
	8	7
6	15	28
✓	✗	✓

```
bool perfect (int x)
```

```
{
```

```
    int i, sum = 0;
```

```
    for (i = 1; i <= x/2; i++)
```

```
        if (x % i == 0)
```

```
            sum += i;
```

```
    if (sum == x)
```

```
        return true;
```

```
    return false;
```

```
}
```

حركة حلوة

لا يمكن لأي رقم أن يكون أحد عوامله أكبر من نصف قيمته.

إذا رأيت كل الناس يسلكون نفس الطريق، عندها فكر قليلاً فقد يكون الطريق الخطأ.

Functions (Prime)

Write a program that reads a positive integer and calls a function that checks if it is prime or not. The main function will print the result.

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

```
bool prime (int);
```

```
int main ( )
```

```
{
```

```
    int x;
```

```
    cout << "Enter a positive integer: ";
```

```
    cin >> x;
```

```
    if (x <= 0)
```

```
        cout << "Invalid number!" << endl;
```

```
    else if (prime (x) == true)
```

```
        cout << "Prime." << endl;
```

```
    else
```

```
        cout << "Not prime." << endl;
```

```
    return 0;
```

```
}
```

Enter a positive integer: **-7**
Invalid number!

Enter a positive integer: **17**
Prime.

Enter a positive integer: **25**
Not prime.

هذا البرنامج كان ميدتيرم
عملي عشرة بالمائة (اشلونك) ؟

الرقم الأولي هو الذي يقبل القسمة
على واحد وعلى نفسه فقط.

ماكو رقم يقبل القسمة على أكثر من نصف قيمته

```
bool prime (int x)
```

```
{
```

```
    for (int i = 2; i <= x/2; i++)
```

```
        if (x % i == 0)
```

```
            return false;
```

```
    return true;
```

```
}
```

إذا لم تضع أنت خطة
لحياتك، فستصبح جزءاً
من خطط غيرك.

Functions (Sum x-to-y)

Write a function that gets three integers and returns the sum of all integers between the first two integers that are divisible by the third one inclusively.

If the integers are: 14 3 5, the function should return 15

If the integers are: 15 21 7, the function should return 21

If the integers are: 12 21 11, the function should return 0

int divisible (int x, int y, int z)

```
{
    int i, min, max, sum = 0;

    if (x < y)
        { min = x; max = y; }
    else
        { min = y; max = x; }

    for (i = min; i <= max; i++)
        if (i % z == 0)
            sum += i;

    return sum;
}
```

inclusively

تعني أن كلا من البداية والنهاية مشمولة
في الأرقام البينية بعكس **exclusively**.

تري البرمجة متعة ! لكن
لمن يحب التفكير (مثلك)؟

أريد أن أكون كل ما
يمكنني أن أكونه.

Functions (*Contains odd?*)

Write a function that gets an integer and returns true if any of its digits is odd.

If the integer is 1236, the function will return *true*.

If the integer is 284, the function will return *false*.

bool digits (int x)

```
{
    int d;

    while (x != 0)
    {
        d = x % 10;
        if (d % 2 != 0)
            return true;

        x /= 10;
    }

    return false;
}
```

إذا كان أحد خانات الرقم odd فإن الدالة ستتوقف على الفور وتعيد true.

يمكن اختصار ما بداخل القوس للتالي:

```
{
    if (x % 2 != 0)
        return true;
    x /= 10;
}
```

أفضل إنجاز هو أن تكون الشخص الذي تريده في عالم يحاول جعلك الشخص الذي لا تريده.

Functions (Average of even digits)

Write a function that gets a positive integer and returns the average of the even digits it contains.

If the integer is 284, the function will return 4.66667

If the integer is 1236, the function will return 4

double digits (int x)

```
{
    int i, d, count = 0, sum = 0;

    while (x != 0)
    {
        d = x % 10;

        if (d % 2 == 0)
        {
            sum += d;
            count ++;
        }

        x /= 10;
    }

    if (count == 0)
        return 0;

    return sum * 1.0 / count;
}
```

إذا أكو تقسيم نجعل النوع double.

تنفيذ (x/10)
بشكل متتالي ينتهي بأي رقم صحيح
إلى الصفر.

قبل إجراء أي عملية تقسيم يجب التأكد
أن المقسوم عليه لا يساوي صفراً.

أنت غير مطالب لتكون أفضل من
أي شخص آخر، أنت مطالب بأن
تكون أفضل شيء يمكنك أن تكونه.

Functions (*Mirror of on integer*)

Write a function that gets a positive integer and returns its mirror.

If the integer is 1236, the function will return 6321

If the integer is 284, the function will return 482

int mirror (int x)

```
{
    int d, m = 0;

    while (x != 0)
    {
        d = x % 10;

        m = m * 10 + d;

        x /= 10;
    }

    return m;
}
```

كلما تدربت على أفكار أكثر ستكون
أفكار الامتحان سهلة وشبه مباشرة
بخلاف التفكير حينها لأول مرة.

إذا ضربت رقم في (10) فكأنك أزحت
خاناته بمقدار خانة إلى اليسار.

حان الوقت لكي نعيش
الحياة التي تخيلناها.

Functions (gcd)

Write a function that gets two positive integers and returns their greatest common divisor.

If the integers are: 12 36, the function will return 12

If the integers are: 24 18, the function will return 6

If the integers are: 15 7, the function will return 1

```
int gcd (int x, int y)
```

```
{
```

```
    int i, min;
```

```
    if (x < y)
```

```
        min = x;
```

```
    else
```

```
        min = y;
```

```
    for (i = min; i >= 1; i--)
```

```
        if (x % i == 0 && y % i == 0)
```

```
            return i;
```

```
}
```

العامل المشترك الأعلى لرقمين
هو أكبر قيمة يقبل الرقمان
القسمة عليه بدون باقي.

إذا ماكو عامل مشترك بين الرقمين
سترد الدالة (1).

ربما لم يعلق الآخرون آمالاً
عريضة علي، ولكن ما زلت
أعلق آمالاً عريضة على نفسي.

Exercise: An application of function floor is rounding a value to the nearest integer, The statement

$y = \text{floor} (x + 0.5);$

rounds the number x to the nearest integer and assigns the result to y.

Write a program that reads several numbers and uses the preceding statement to round each of these numbers to the nearest integer.

(Hint: use <cmath> functions).

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

```
int main ( )
```

```
{
```

```
    double x, y;
```

```
    cout << "enter number (-1 to exit): ";
```

```
    cin >> x;
```

```
    while ( x != -1 )
```

```
    {
```

```
        y = floor ( x + 0.5 );
```

```
        cout << x << " rounded to the nearest integer is: " << y << endl;
```

```
        cout << "enter number (-1 to exit): ";
```

```
        cin >> x;
```

```
    }
```

```
    return 0;
```

```
}
```

تذكر أننا نستخدم while عند قراءة عدة أرقام غير محدد عددها.

اقرأ مرة قبل loop ومرة داخل ال loop بنهايتها . { شابتر (2) }

```
enter number (-1 to exit): 3.2
```

```
3.2 rounded to the nearest integer is: 3
```

```
enter number (-1 to exit): 4.6
```

```
4.6 rounded to the nearest integer is: 5
```

```
enter number (-1 to exit): 10
```

```
10 rounded to the nearest integer is: 10
```

```
enter number (-1 to exit): -1
```

قدراتنا مثل دراجة تتمتع بعشر سرعات، لكن أغلبنا يركن إلى السرعة الأقل.

Exercise: Define a function *hypotenuse* that calculates the length of the hypotenuse of a right triangle when the other two sides are given. Use this function in a program to determine the length of the hypotenuse for each of the triangles below:

side1 = 3.0	side2 = 4.0
side1 = 5.0	side2 = 12.0
side1 = 8.0	side2 = 15.0

```
# include <iostream>
# include <iomanip>
# include <cmath>
using namespace std;
```

من نظرية فيثاغورث

$$hyp = \sqrt{(side1)^2 + (side2)^2}$$

```
double hyp ( double, double );
```

```
int main ( )
{
    cout << setw(5) << "Side1" << setw(7) << "Side2" << setw(12) <<
        "Hypotenuse" << endl;
    cout << setprecision(1) << fixed;

    cout << setw(5) << 3.0 << setw(7) << 4.0 << setw(12) <<
        hyp ( 3.0, 4.0 ) << endl;
    cout << setw(5) << 5.0 << setw(7) << 12.0 << setw(12) <<
        hyp ( 5.0, 12.0 ) << endl;
    cout << setw(5) << 8.0 << setw(7) << 15.0 << setw(12) <<
        hyp ( 8.0, 15.0 ) << endl;
    return 0;
}
```

```
double hyp ( double x, double y )
{
    return sqrt ( x * x + y * y );
}
```

Side1	Side2	Hypotenuse
3.0	4.0	5.0
5.0	12.0	13.0
8.0	15.0	17.0

قسم أي مهمة صعبة إلى مهام أصغر
أسهل فإن الجبال تتكون من الحصى.

Exercise: Write a function that takes a number and a character, and prints a solid square of that character with side equal to the number. For example, if 4 and # are read, the following square is printed.

```
####
####
####
####
```

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

```
void square ( int, char );
```

```
int main ( )
{
    int size;
    char c;

    cout << "Enter the size: ";
    cin >> size;
    cout << "Enter a character: ";
    cin >> c;

    square ( size, c );

    return 0;
}
```

```
void square ( int size, char c )
{
    for ( int i = 1; i <= size ; i++ )
    {
        for ( int j = 1; j <= size; j++ )
            cout << c;

        cout << endl;
    }
}
```

```
Enter the size: 5
Enter a character: *
*****
*****
*****
*****
*****
```

لا يعدم الفاشل اختلاق
ألف عذر ليبرر بها فشله.

Exercise: Write a program that inputs three double numbers and passes them to a function that returns the smallest number.

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

double smallest ( double, double, double );

int main ( )
{
    double x, y, z, min;

    cout << "Enter three numbers: ";
    cin >> x >> y >> z;

    min = smallest ( x, y, z );

    cout << "The smallest number is: " << min << endl;

    return 0;
}

double smallest ( double x, double y, double z )
{
    if ( x < y && x < z )
        return x;
    if ( y < z )
        return y;
    return z;
}
```

Enter three numbers: 4 10 2
The smallest number is: 2

لتكن أنت أهم بند في
قائمة اهتماماتك.

Exercise: Write a function *distance* that calculates the distance between two points (x1, y1) and (x2, y2). All numbers and return values are of type double.

```
# include <iostream>
# include <cmath>
# include <iomanip>
using namespace std;
```

```
double distance ( double, double, double, double );
```

```
int main ( )
{
    double x1, x2, y1, y2, d;

    cout << "Enter the first point: ";
    cin >> x1 >> y1;
    cout << "Enter the second point: ";
    cin >> x2 >> y2;

    d = distance (x1, y1, x2, y2 );

    cout << setprecision(2) << fixed;
    cout << "The distance between the two points is: " << d << endl;

    return 0;
}
```

```
double distance ( double x1, double y1, double x2, double y2 )
{
    return sqrt ( pow ( x2 - x1, 2 ) + pow ( y2 - y1, 2 ) );
}
```

Enter the first point: 3 4
 Enter the second point: 5 6
 The distance between the two points is: 2.83

لك شيء في هذا العالم،
فهيأ لتحصل عليه.