

# Comp 133

BY: MOAYAD ALMASRI

```
using System;
using System.Data;

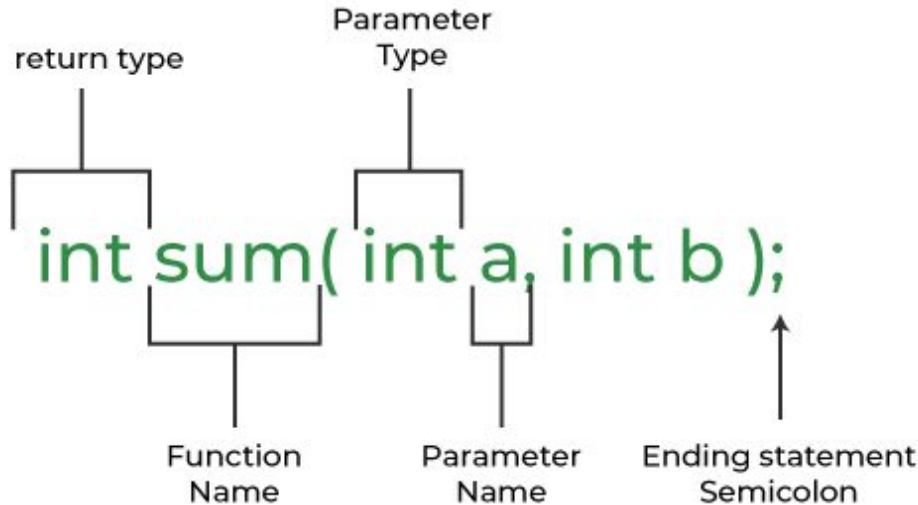
namespace MyProject
{
    /// <summary>
    /// Customer Master
    /// </summary>
    /// <remarks> </remarks>
    public class Customer
    {
        private string pCustomerCode;
        private string pCustomerName;
        private string pAddress;
        private string pMobileNo;
        private string pEmail;

        public string CustomerCode
        {
            get { return pCustomerCode; }
            set { pCustomerCode = value; }
        }

        public string CustomerName
        {
            get { return pCustomerName; }
            set { pCustomerName = value; }
        }

        public string Add
        {
            get { return pAddress; }
            set { pAddress = value; }
        }
    }
}
```

# function



يجب ان يكون الناتج النهائي ل  
**function** من نفس نوع ال**function**  
, أي انه اذا كان الفنكشن (int) فان  
ال**return** سوف يكون **int** , وهكذا مع  
جميع انواع ال**function** , الا ال**void**  
فانها لا تسترجع شئ

# function

```
double cours() {  
    double a=3.1;  
    return a;  
}
```

هنا تم إنشاء function من نوع double وجعله يسترجع قيمة  
double

# function

```
int function() {  
    return 3.0;  
}
```

هنا سوف يكون لدينا error بسبب اننا عرفنا  
function من نوع int وجعلها تسترجع  
double

# function

ولكن ماذا لو كان لدينا 2 function يملكان نفس الاسم ؟

```
int function() {  
    return 3;  
}
```

```
double function() {  
    return 3 ;  
}
```

في هذا المثال تم إنشاء 2 function يملكان نفس الاسم ونفس النوع من ال parameter ( الاثنين no argument ) ولهذا سوف يحصل معنا **error** ولا يمكن أن يتشابه

function 2 بالاسم والparameter

# function

## void function

النوع الثاني من function والذي لا يحتاج إلى return هو ال void function, وهذا function يستخدم للقيام بعمليات مختلفة على سبيل المثال اريد ان اخرج واجهة يراها المستخدم اقوم بإنشاء function تحتوي على المعلومات التي سوف يتم اعطائها للمستخدم واستدعاء ال function وقت الحاجة لها

# function

قمنا بإنشاء void function لحساب قيمة عددين وطباعة الناتج على  
صيغة  $x+y=sum$

```
void sumation(int x , int y) {  
    int sum = x+y ;  
    printf("%d",sum) ;  
}
```

# function

```
void sumation(int x , int y){  
    int sum = x+y ;  
    printf("%d",sum);  
}
```

```
int i = 4 ;  
int j = 5 ;  
sumation(i , j);
```

نقوم بإستدعاء function  
واعطائه قيمة z , i فتكون  
المخرجات :  
 $9=4+5$



Every function  
prototype must  
include at least one  
parameter.

- a) true
- b) false



What is the output after executing the following code fragment

- a)5
- b)0
- c)6
- d)none

```
void f(int x)
{ x = x + 5; }
int main()
{
    int x = 5;
    f(x);
    printf("%d", ++x);
}
```

# The loop

01

While loop

- `while (x==y){`  
`}`

02

for loop

- `for(int i = x ; i < y ; i++){`  
• `}`

03

do-while loop

- `do{`  
• `} while(x==y) ;`

# The loop

02

for loop

- `for(int i = x ; i < y ; i++){`
- `}`

# The loop

تعريف المتغير أو إحضاره وإعطائه قيمة ابتدائية

02

for loop

- `for(int i = x ; i < y ; i++){`
- `}`

# The loop

يتم الفصل بين اجزاء ال for عن طريق ال ; يرجى  
الانتباه وعدم وضع : لانه سوف يسبب error

02

for loop

- `for(int i = x ; i < y ; i++){`
- `}`

# The loop

الشرط الذي سوف يتم وضعه لاستمرار عمل الحلقة

02

for loop

- `for(int i = x ; i < y ; i++){`
- `}`

# The loop

## Loop inside loop (nested loop)

- `for(int i = x ; i < y ; i++){`
- `for(int j = x ; j < y ; j++){`
- `}`
- `}`

عندما يتم عمل loop في داخل ال loop نقوم بتكرير الحلقة الداخلية في كل مرة تتكرر فيها الحلقة الخارجية وتعود من الصفر



# The loop

ما مخرجات هذا الكود ؟

```
for (int i = 1; i < 3; i++) {  
    for (int i = 1; i <= 2; i++) {  
        printf("hi");  
    }  
}
```

# The loop error

```
for (int i = 1; i < 3; i++) {  
    for (int i = 1; i <= 2; i++) {  
        printf("hi");  
    }  
}
```

# The loop

ما مخرجات هذا الكود؟

```
for (int i = 1; i < 3; i++) {  
    for (int j = 1; j <= 2; j++) {  
        printf("hi");  
    }  
}
```

# The loop

ما مخرجات هذا الكود ؟

```
for (int i = 1; i < 3; i++) {  
    printf("hello");  
    for (int j = 1; j <= 2; j++) {  
        printf("hi");  
    }  
}
```

# The loop

نبدأ الحلقة بقيمة ابتدائية ل  $i = 1$  , وعندما ندخل الحلقة نطبع كلمة hello

output

```
for (int i = 1; i < 3; i++) {  
    printf("hello");  
    for (int j = 1; j <= 2; j++) {  
        printf("hi");  
    }  
}
```

hello

# The loop

الان ندخل بالحلقة الثانية حيث سوف يتم تنفيذها والانتهاء منها ثم اكمال  
الحلقة الخارجية

output

```
for (int i = 1; i < 3; i++) {  
    printf("hello");  
    for (int j = 1; j <= 2; j++) {  
        printf("hi");  
    }  
}
```

hello

# The loop

الحلقة تبدأ بقيمة 1 فتقوم بطباعة hi ثم نقوم بالزيادة ولأنه يوجد مساواة بالشرط فسوف يتم تنفيذه عندما تكون قيمة  $j=2$  , فيقوم بطباعة hi مرتان

output

```
for (int i = 1; i < 3; i++) {  
    printf("hello");  
    for (int j = 1; j <= 2; j++) {  
        printf("hi");  
    }  
}
```

hello

hi

hi

# The loop

تقوم الحلقة باعادة نفسها بعد ان اصبحت قيمة  $i=2$  وتقوم بطباعة hello  
وتم ندخل الحلقة الداخلية وتعيد نفسها من الصفر وتكرر العمليات السابقة

output

سوف تتوقف الحلقة  
عند الدورة الثانية لها  
بسبب تجاوز الشرط  
ولانه لا يوجد مساواة  
عند ال3 فتتوقف  
الحلقة ويكون الناتج  
كما هو مبين عاليمين

```
for (int i = 1; i < 3; i++) {  
    printf("hello");  
    for (int j = 1; j <= 2; j++) {  
        printf("hi");  
    }  
}
```

hello

hi  
hi

hello  
hi  
hi



How many times is the printf statement executed?

A)21

**b)15**

c)10

d)none

```
for(int i=0; i<6; i++)
```

I=0,1,2,3,4,5

```
for(int j=0; j<i; j++)
```

```
printf("%d ", j * i);
```

I=1

J=0

1\*0

I=2

J=0,1

2\*0

2\*1

I=3

J=0,1,2

3\*0

3\*1

3\*2

I=4

J=0,1,2,3

4\*0

4\*1

4\*2

4\*3

I=5:

j=0,1,2,3,4

:

5\*0

5\*1

5\*2

5\*3

5\*4

How many times will the following loop run?

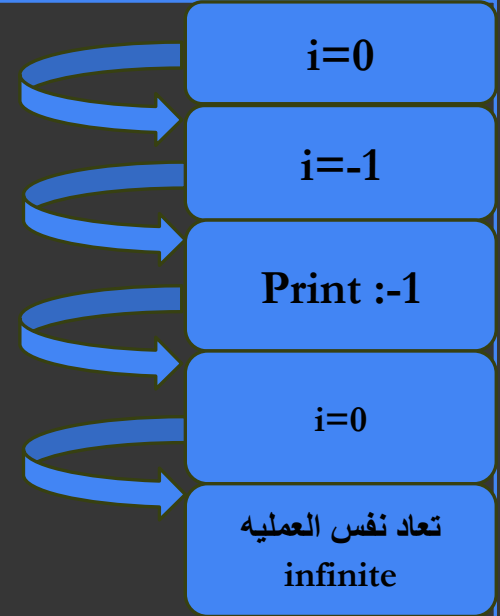
A)0

B)1

C)infinite

D)compiler error

```
int i = 0;  
do  
{  
    printf("%d", --i);  
    i++;  
}while (i >= 0);
```



How many times the statement `printf("\n")` will get executed?

- a)5
- b)15
- c)2**
- d)none

```
for (int i = 0; i < 15; ++i)
{
    printf("%d", i++);
    if (i % 5 == 0)
        break;
    printf("\n");
}
```

What the output?

a)5

b)2

c)6

d)none

```
int a=5,c=1;
```

```
Do{
```

```
C++;
```

```
}while(--a>0);
```

```
printf("%d",c);
```

How many times will the following loop be executed??

- a)0
- b)100
- c)infinite
- d)1

```
int x = 1;  
while (x <= 100)  
{  
    printf("%d ", x--);  
}
```

# if & switch

01

**If**

- `if(x==y){`
- `}`

02

**switch**

- `switch(x){`
- `Case 1 :`
- `break ;`
- `}`

03

**else - if**

- `if(x==1){`
- `} else if (x==2){`
- `} else{`
- `}`

# break & Continue

**Break** : تقوم بالخروج من الحلقة وعدم تنفيذ الأسطر التي تليها في حالة تحقق الشرط

**Continue** : تقوم باكمال الحلقة ولكنها تمنع تنفيذ ما تحتها في حالة تحقق الشرط

# break & Continue

```
int i = 1 ;  
while (++i < 8){  
    printf("palestine");  
    if(i==4) {  
        break;  
    }  
    printf("%d",i);  
}  
printf("berzait");
```



# break & Continue

```
int i = 1 ;  
while (++i < 8){  
    printf("palestine");  
    if(i==4) {  
        break;  
    }  
    printf("%d", i);  
}  
printf("berzait");
```

The output is :

palestine  
2  
palestine  
3  
palestine  
berzait

# break & Continue

```
int i = 1 ;  
while (++i < 8){  
    printf("palestine");  
    if(i==4) {  
        continue;  
    }  
    printf("%d",i);  
}  
printf("berzait");
```

# break & Continue

```
int i = 1 ;  
while (++i < 8){  
    printf("palestine");  
    if(i==4) {  
        continue;  
    }  
    printf("%d",i);  
}  
printf("berzait");
```

The output is :

palestine  
2  
palestine  
3  
palestine  
palestine  
5  
palestine  
6  
palestine  
7  
berzait

The output of the following code segment is:

- A)Hello
- B)Hi
- C)HelloHi
- d)None

```
int x = 1
switch(x)
{ case 1:
  printf("Hello");
default:
  printf(" Hi");
}
```

بسبب عدم وجود  
**Break**  
لهذا تقوم بتنفيذ الجملتين  
**Output**  
**is :Hello**  
**Hi**

What is the value of balance after the following code is executed?

**A)1**

B)infinte loop

C)-1

D)none

```
int balance = 10;
while (balance > 1) {
    if (balance < 9) {
        continue;
    }
    balance = balance - 9; }
```

عندما يكون

Balance

اكثر من 1 سوف يدخل  
الحلقه

ووظيفه الجمله الشرطيه

If

هو تخطي (تجاهل) كل  
الارقام التي اصغر

من 9 هكذا يبقى فقط

Balance=10

هو مخرج الحلقه

Balance=10-9=1

he output of the following code segment is:

A)8

B)6

D)0

C)none

```
int x = 6;  
for(; x > 1; x += 2)  
if(x % 3)  
break;  
printf("%d", x);
```

طالما قيمه المتغير اكبر من 1 سيدخل الحلقه وهنا دخل الحلقه لان قيمته 6 بعدها يفحص اذ كان الشرط يتحقق اذا تحقق الشرط ستنتهي الحلقه واذا لم يتحقق سيضاف للمتغير 2

الرقم 6 باقي قسمته على 3 هو 0 والذي يعني

$(False) 0 = 6 \% 3$

اي لم يتحقق الشرط وهنا سيضاف الرقم 2 للمتغير

$8 = 6 + 2$

الرقم 8 باقي قسمته على 3 هي 1

$(true) 1 = 8 \% 3$

سيدخل الحلقه ويحقق الشرط الذي يوقف الحلقه

WHAT THE VALUE OF N  
AFTER EXECUTING THE  
FOLLOWING CODE:

A)2

B)5

C)7

D)NONE

```
int n=1;  
if(n)  
{ n+=3; // n = 1 + 3 = 4  
  n/=2; // n = 4 / 2 = 2  
}  
Else  
n+=4;  
n=7;  
printf("%d", n);
```

غير تابعه للجمله  
الشرطيه  
بسبب عدم وجود  
{  
بعد  
else

In following code, how many times the value of n will be printed on screen:

- A) 4 times
- B) 5 times**
- c) 3 times
- d) 10 times

```
int n;  
for(n=1;n<=10;n++)  
printf("%d\n",++n);
```

N=2

N=4

N=6

N=8

N=10



What is the result after the execution of the following code if **a=10**, **b=5** and **c=10**?

A) a=5, c=6

B) a=10, c=11

C) none

D) a=11, c=10

```
if ((a > b) && (a < c))
```

```
    a = a + 1;
```

```
else
```

```
    c = c + 1;
```

C=c+1=11

T && f = f

a=10

What the output of following code

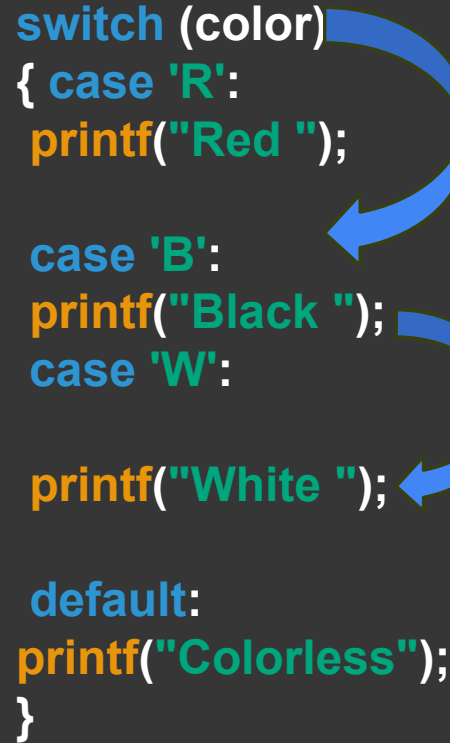
- a)Red
- b)Black
- c)Black White
- d)Black White Colorless

```
switch (color)
{ case 'R':
  printf("Red ");

  case 'B':
  printf("Black ");
  case 'W':

  printf("White ");

  default:
  printf("Colorless");
}
```



سبب تنفيذ الجمل بعد  
Case 'B':  
Printf("Black")  
هو عدم وضع بعدها  
Break;

The output is

A)ok 83

B)ok 8

C)ok 80

D)none

```
int main()
{
    int z = 83; if (z >= 90 && z <= 100) F&&T
    {
        printf("great\n");
    }
    else if (z < 60) f
    {
        printf("fail\n");
    }
    else if (z >= 60 && z < 90) T&&T
    {
        printf("ok %d\n", z / 10 * 10);
    }
```

Int/int=int

The output is

A) 5 30

B) 6 25

C) 5 25

D) 5 24

```
int x = 3, n = 6;
```

```
if (x = 2)
```

```
    x += 2;
```

```
else
```

```
    x *= 2;
```

```
    x *= n--;
```

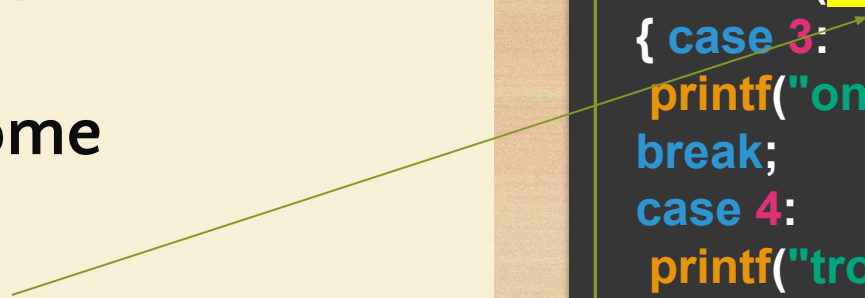
```
Printf("%d %d", n, x);
```



The output is:

- a)go home
- b)tow
- c)error
- d)none

```
int x = 14;  
switch (x / 3.0)  
{ case 3:  
  printf("one");  
  break;  
  case 4:  
    printf("tro");  
    break;  
  case 5:  
    printf("hi ");  
    break;  
  default:  
    printf("go home"); }
```



The output is:

- a)bad
- b)nice bad
- c)nice
- d)none

```
int x = 10, y = 23, z = 15;  
if (x % y > 0 && z / 15 || 15 % 2)  
    printf("nice.");  
Else  
    printf("bad ");
```

Which kind  
of statement can be  
nested inside the body  
of an if statement:

- a) if statement
- b) for statement
- c) while statement
- d) all of above

```
isVideo = ( type == "image" ) || ( $.isFunction( source.isImage ) )  
isUrl = ( type == "url" ) || ( $.isFunction( source.isUrl ) )  
isElement = ( type == "element" ) || ( $.isFunction( source.isElement ) )  
isObject = ( typeof subject == "object" ) || ( $.isFunction( source.isObject ) )  
  
// Check if boxer is already active, return default  
if ( $("#boxer").length > 1 || ( !isImage || !isVideo ) )  
    return;  
}  
  
// Kill event  
_killEvent(e);  
  
// Cache internal data  
data = $.extend( {}, {  
    $window: $(window),  
    $body: $("body"),  
    $target: $target,  
    $object: $object,  
    visible: false,  
    resizeTimer: null,  
    touchTimer: null,  
    gallery: {  
        active: false  
    }  
}, e.data.namespace,
```