نکته: درهرزمان می‌توان با استفاده از دستورات زیر از حلقه خارج شد:

Exit for ⟶ for ساختار

Exit Do ⟶ { Do-while / Do-until

Exit while ⟶ while ساختار

نکته: می‌توان با استفاده از دستور Continue اجرای دستورات حلقه را قطع و آن را از ابتدای حلقه از سرگرفت.

مثال: با استفاده از دستور Continue ؛ اعداد فرد میان ۱ تا ۱۰۰ را در خروجی نمایش دهید؟

```
Dim i As integer = 1
for i = 1 to 100 Step 1
    if (i Mod 2 = 0) Then
        Continue for
    else
        Console.writeline(i.ToString)
    endif
next
```

نکته: از دستور Continue می‌توان در حلقه‌های while ، Do-while هم استفاده کرد.

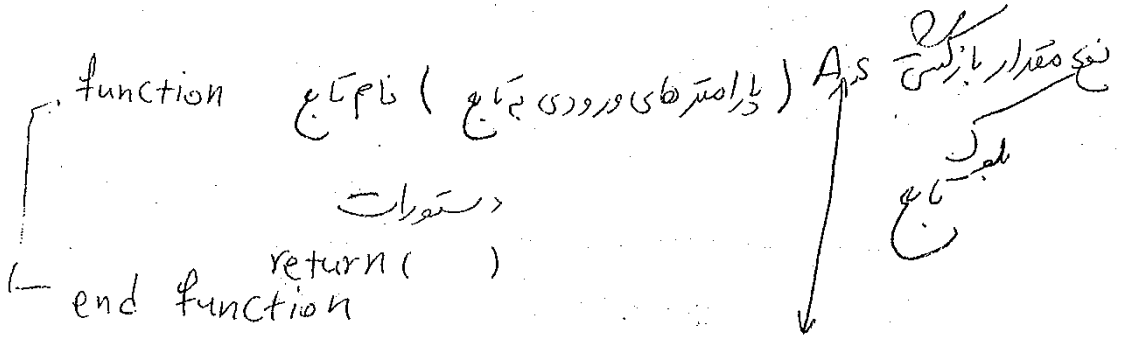
Continue for ⟶ for ساختار

Continue Do ⟶ { Do-while / Do-until

Continue while ⟶ while ساختار

گاهی مجموعه‌ای از دستورات در بخش‌های مختلف برنامه به صورت تکراری استفاده می‌شوند. در این حالت به جای تکرار می‌توان این مجموعه را در بخشی از برنامه نوشته و بارها مورد استفاده قرار داد. این روش در برنامه‌های بزرگ و پیچیده به خوانا کردن و قابل فهم کردن برنامه و کاهش زمان لازم برای خطایابی و خطا زدایی می‌انجامد.

1- تابع: مجموعه‌ای از دستورات که عملیاتی را انجام داده و مقداری را برمی‌گرداند.



1- نام تابع از قواعد مربوط به نامگذاری متغیرها پیروی می‌کند

2- درست مربوط به پارامترها، تام و نوع متغیرهای ورودی به تابع را مشخص کنید

3- تعداد و نوع پارامترهای ورودی به تابع در حکم امضای تابع هستند.

4- با استفاده از دستور return تابع مقداری را بازگشت می‌دهد.

5- برای فراخوانی تابع در داخل برنامه از عبارت زیر استفاده کنید.

آرگومان‌های پارامترها = نام تابع = نام متغیر

6-

مثال: تابعی بنویسید که طول و عرض یک مستطیل را گرفته و مساحت آن را محاسبه کرده و بازگشت دهد؟

۱- ابتدا در خارج از ساب روتین Main که نقطه‌ی شروع اجرای برنامه ها در محیط Console است تابع را به شکل زیر تعریف می کنیم:

```
function Area(ByVal L1 As Single, _
              ByVal L2 As Single) As Single
    Dim r As Single
    r = L1 * L2
    return(r)
end function
```

underline: برای قطع دستور و نوشتن ادامه‌ی دستور جاری در سطر بعد؛

2- در ساب روتین Main برای فراخوانی تابع به شکل زیر عمل می کنیم:

```
Sub Main()
    Dim a As Single
    Dim b,c As Single
    a = CType(Console.Readline(), Single)
    b = CType(Console.Readline(), Single)
    c = Area(a,b)
    Console.writeline(c.ToString)
End Sub
```

برای تبدیل متغیر ورودی به نوع Single

فراخوانی تابع

برای تبدیل به رشته

2- Sub (سابروتین) : شبیه به تابع است اما هیچ مقداری را بازگشت نمی دهد.

```
Sub نام سابروتین (پارامترهای ورودی)
    دستورات
End Sub
```

1- نام سابروتین از قواعد مربوط به نامگذاری متغیرها پیروی می کند

2- در قسمت مربوط به پارامترها، نام و نوع متغیرهای ورودی به سابروتین را مشخص کنید

3- برای فراخوانی سابروتین از دستور زیر استفاده کنید :

نام سابروتین (آرگومان های پاس شده)

مثال: سابرویتی بنویسید که طول و عرض یک مستطیل را گرفته و مساحت آن را حساب کرده و خروجی نمایش دهد؟

۱- ابتدا در خارج از سابرویتی Main یک سابرویتی مانند را به شکل زیر تعریف می‌کنید:

```
Sub Area (ByVal L1 As Single, ByVal L2 As Single)
    Dim r As Single
    r = L1 * L2
    Console.WriteLine(r.ToString)
End Sub
```

2- در سابرویتی Main برای فراخوانی سابرویتی Area به شکل زیر عمل می‌کنید:

```
Sub Main()
    Dim a As Single = 1.1
    Dim b As Single = 2.2
    Area(a, b)  ⟵ فراخوانی سابرویتی
End Sub
```

ارسال آرگومان‌ها :

برای ارسال آرگومان‌ها به توابع و ساب‌روتین‌ها دو روش وجود دارد :

1- با مقدار 2- با ارجاع

1- با مقدار (Byval) : به صورت پیش‌فرض همه آرگومان‌ها به صورت Byval ارسال می‌شوند (یا می‌گوئیم) :

هرگاه آرگومانی به شکل Byval ارسال شود یک کپی از آن تهیه و در اختیار تابع یا ساب‌روتین قرار می‌گیرد. تابع یا ساب‌روتین، تغییر خود را بروی آن اعمال داده و متغیر اصلی دست نخورده و پس از پایان تابع یا ساب‌روتین، نسخه‌ی کپی از بین می‌رود.

مثال زیر کدهای توضیحات بالاست :

```
Sub P (Byval n As Integer)
    n = n + 1
    Console.writeline(n)
endSub

Sub Main( )
    Dim n As integer = 5
    Console.writeline(n)
    P(n)        ← ارسال با مقدار
    console.writeline(n.tostring)
end Sub
```

خروجی به شکل زیر است :

1	ط	5
2	ط	6
3	ط	5

2- با ارجاع (ByRef):

برخلاف ByVal در ByRef، یک ارجاع (آدرس) حافظه‌ای که محل واقعی متغیر در حافظه را نشان می‌دهد) به تابع یا ساب‌روتین ارسال می‌گردد. بنابراین وقتی که متغیر پارامتر در درون ساب‌روتین یا تابع تغییری کند، در واقع نحوه‌ی اصلی را تغییری کند. به این ترتیب پس از خروج از تابع یا ساب‌روتین، مقدار متغیر ارسالی، منعکس کننده تغییری است که در تابع یا ساب‌روتین روی آن صورت گرفته است.

مثال زیر گویای توضیحات است:

```
Sub P(ByRef n As integer)
    n += 1
    Console.writeline(n)
endsub

Sub Main()
    Dim n As integer = 5
    Console.writeline(n.toString)
    P(n) ⟵ ارسال با ارجاع
    Console.writeline(CStr(n))
endsub
```

خروجی به صورت زیر است:

1 سطر	5
2 سطر	6
3 سطر	6

پارامترها ورودی را می توان به دو شکل زیر تعریف کرد:

1- اجباری 2- اختیاری

1- اجباری: در این حالت هنگام فراخوانی تابع یا ساب پروسیجر باید مقداری متناظر با پارامتر ورودی، به تابع یا ساب پروسیجر ارسال کرد.

2- اختیاری (Optional): در این حالت هنگام فراخوانی تابع یا ساب پروسیجر ارسال مقدار اختیاری است و اگر مقداری ارسال نشود مقدار پیش فرض که در هنگام تعریف پارامترها در نظر گرفته می شود؛ مورد محاسبه قرار می گیرد. هر پارامتر اختیاری باید دارای یک مقدار پیش فرض باشد. پارامترهای اختیاری باید آخرین پارامترهای یک تابع یا ساب پروسیجر باشند. تعداد این پارامترها محدودیتی ندارد اما بعد از آن ها پارامتر اجباری نمی توان تعریف کرد.

در مثال زیر یک ساب پروسیجر تعریف می شود و در قسمت Main به اشکال مختلف مورد فراخوانی قرار می گیرد:

```
Sub P(ByVal a As Integer, Optional ByVal b _
        As integer=5)
    Console.writeline (a+b)
End Sub

Sub Main( )
    P(6,6) ⟶ در خروجی مقدار 12 چاپ می شود
    P(6) ⟶ در خروجی مقدار 11 چاپ می شود
End Sub
```

ارسال آرگومانها به تعداد متغیر:

برای این کار از کلیدی ParamArray استفاده می‌کنیم که در این حالت آرگومانهایی که بعد از آن می‌آیند به صورت یک آرایه به تابع یا ساب‌روتین ارسال می‌شوند

```
Sub نام ساب‌روتین (ByVal ParamArray نام متغیر() As نوع متغیر)
    دستورات
End Sub
```

```
Function نام تابع (ByVal ParamArray نام متغیر() As نوع متغیر) As نوع مقدار بازگشتی
    دستورات
End Function
```

مثال: ساب‌روتینی بنویسید که به تعداد متغیر، از ورودی عدد بگیرد و حاصلجمع آنها را محاسبه کرده و در خروجی نمایش دهد؟

```
Sub Sum(ByVal ParamArray a() As Integer)
    Dim r As Integer = 0
    For i As Integer = a.GetLowerBound(0) To
        a.GetUpperBound(0) Step 1
            r = r + CStr(a(i))
    Next i
    Console.WriteLine(r)
End Sub
```

حفظ مقدار داده ها مانند حاصل توابع فراخوانی تابع یا ساب روتین، برای نگهداری مقدار داده ها از کلمه ی کلیدی Static استفاده می کنیم.

Static نام متغیر As نوع متغیر

مثال زیر را به دقت بررسی کنید:

```
Sub a()
    Dim b As Integer=0 → مقداردهی اولیه
    b = b + 1
    Console.writeline(b.tostring)
end sub

Sub c()
    Static d As integer=0
    d = d + 1
    Console.writeline(d.tostring)
end sub

Sub Main()
    a()
    a()
    a()
    c()
    c()
    c()
end sub
```

خروجی برنامه به شکل زیر است:

1 } a()
1
1

1 } c()
2
3