برخی از فضاهای نام مهم عبارتند از:

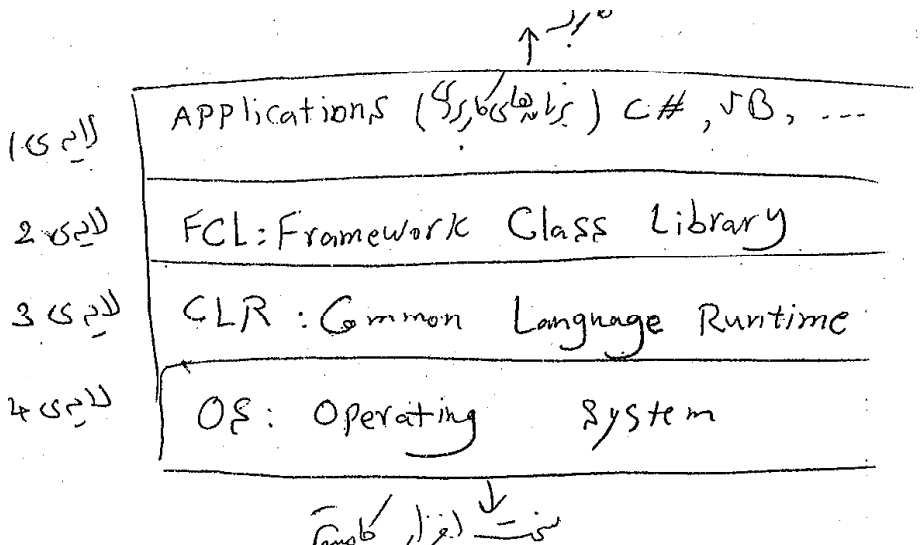
1- System : حاوی کلاس های پایه ای است.

2- System.Collection : حاوی کلاسهای کار با انواع کلکسیون ها مثل لیست آرایی و جدول دیکشنری می باشد.

3- System.Windows.Forms : حاوی کلاسهایی برای ساخت برنامه های کاربردی ویندوزی است. با استفاده از این کلاسها می توان با به کارگیری امکانات سیستم عامل ویندوز، واسطه های کاربری غنی ایجاد کرد.

4- System.Data : حاوی کلاسهای لازم برای کار بردی با داده ها و دیتابیس ها مثل SQL Server می باشد.

5-



با توجه به شکل بالا یک معماری لایه ای را می توان در نظر گرفت.

1- در لایه ی اول برنامه های کاربردی قرار دارند که با زبانهایی مثل C# و VB نوشته می شوند.

2- در لایه ی دوم کتابخانه کلاس های چهارچوب .Net یا FCL قرار دارد. این کتابخانه ها در DLL هایی قرار دارند که به آنها Assembly گفته می شود. به منظور افزایش سرعت اجرای برنامه های مبتنی بر .Net، این اسمبلی ها تنها هنگام نیاز Load می شوند. با توجه به تعداد زیاد کلاسهای موجود در چهارچوب .Net، مایکروسافت آنها را در NameSpace ها دسته بندی می کند. کلاسها بر حسب نوع کاری که انجام می دهند در NameSpace ها دسته بندی می شوند.

3- در لایه سوم CLR قرار دارد. CLR قلب .NetFramework است. CLR مدیریت اجرای دستورات را بعهده دارد و سرویس های لازم مانند امنیت، مدیریت حافظه، هماهنگی حذف زباله را در اختیار برنامه های کاربردی قرار می دهد.

Data Type :

انواع داده‌ها در VB.Net به دو گروه زیر تقسیم می‌شوند:

1- داخلی : که به صورت پیش‌ساخته در درون زبان قرار دارند. مثل
Integer - Char - String - Short , ...

2- کاربرساز یا UDT : که توسط برنامه‌نویس تعریف می‌شوند. مثل
Class - Structure - Enum , ...

انواع داده‌ها در VB.Net عبارتند از:

Type	Size (Byte)	
Boolean	N/A	True/False
Byte	1	
Short	2	اعداد صحیح
Integer	4	~
Long	8	~
Single	4	اعداد اعشاری
Double	8	~
Char	2	کاراکتر
String	N/A	رشته
Date	8	متغیر تاریخ و زمان
Object	4	هر مقداری

در VB.Net انواع داده‌ها برحسب نحوه ذخیره‌سازی و نحوه دسترسی به دو گروه زیر تقسیم می‌شوند:

1- مقداری: این دسته مقادیر مربوط به خود را درون خود نگه داری می‌کنند و تنها بر این می‌توان مقدار آن‌ها را مستقیماً عمل دستیابی قرار داد.
مثل: Integer, Single, ...

2- ارجاعی: این دسته مقادیر مربوط به خود را نگه نمی‌دارند و به جای آن آدرس محل واقعی داده‌ها را در خود جای می‌دهند. بنابراین برای دسترسی به داده‌های واقعی در این دسته ابتدا باید آدرس محل آن‌ها را به دست آورد و سپس با استفاده از آن به سراغ داده‌های واقعی رفت. مثل:
String, Object, ...

متغیر: قسمتی از حافظه برای ذخیره اطلاعات است.

Dim نام متغیر As نوع متغیر → ذاتی / کاربرساز

1- نام متغیر باید با کاراکتری از میان کاراکترهای a تا z یا A تا Z شروع شود

2- نام متغیر می‌تواند حاوی اعداد 0 تا 9 باشد

3- نام متغیر می‌تواند با underline شروع شود و یا حاوی آن باشد.

4- اگر بخواهیم کامپایلر برنامه‌نویس را مجبور به تعریف متغیر به صورت زیر کند باید از Option Explicit ON استفاده کنیم.

Dim نام متغیر

5- اگر بخواهیم کامپایلر برنامه‌نویس را مجبور به تعریف متغیر به صورت زیر کند باید از Option Strict ON استفاده کنیم.

Dim نام متغیر As نوع متغیر

6- بهتر است همواره نوع داده‌های هم متغیرها را مشخص کنید تا کامپایلر بتواند موارد استفاده نامناسب آنها را تشخیص و تبدیل آنها به یکدیگر را با سرعت بیشتری انجام دهد. در غیر اینصورت بسیاری از اشکالات تا زمان اجرای برنامه معلوم نخواهد شد و بنابراین احتمال آنکه برنامه در زمان اجرا، به دلیل وقوع چنین خطاهایی متوقف شود افزایش می‌یابد. ضمناً سرعت تبدیل انواع داده‌های مختلف به یکدیگر نیز کاهش می‌یابد که ممکن است باعث کاهش سرعت ملی برنامه شود.

چند مثال از تعریف متغیر:

```
Dim a , b As Short          ⟶   تعریف دو متغیر از نوع short
Dim a As Short , b , c As Integer ⟶
Dim a As Integer = 100      ⟶   مقداردهی اولیه به متغیر a
```

ثابت ها:

ثابت هم مثل متغیر، یک مقدار را در خود ذخیره می‌کند اما برخلاف متغیر مقدار موجود در یک ثابت را نمی‌توان در جریان اجرای برنامه تغییر داد.

Const نام ثابت As نوع ثابت ← داده
↘ کاربر ساز

۱- نام یک ثابت از قوانین مربوط به نام گذاری متغیرها پیروی می‌کند.

مثال: برنامه‌ای بنویسید که مساحت دایره را بدست بیاورد؟

Const Pi As Double = 3.14 ← عددی

Dim r As Double

Dim a As Double = Pi * r ^ 2

a ← مساحت دایره — r ← شعاع دایره

انواع ثابت: ۱- نامدار (Symbolic) ۲- literal ۳- شمارشی (enum)

۱- نامدار: با کلمه کلیدی Const شروع می‌شود ←

Const b As Integer = 101

۲- literal: مثل عدد 32 در دستور مقابل ← a = a + 32

۳- شمارشی (enum): کاهی کارکردن با ثابت های نامدار (شمارشی) آسانتر از کارکردن با مقادیر ثابت عددی است. با استفاده ثابت شمارشی می‌توان ثابت‌ها نامدار مرتبط ایجاد کرد. چنین مجموعه‌ای می‌تواند حاوی نام رنگها یا حتی نام روزهای هفته باشد.

enum نام ثابت

عنصر 1
عنصر 2
⋮
عنصر n

end enum

بلوک enum

↓↓

enum Color
    Red = 11
    Green = 1
    Blue
endenum

→ Dim c As Color
متغیر از

c = Color.Red

یا

c = 11 → در صورتی که Option Strict Off باشد

c = CType(11, Color) →

در صورتی که Option Strict ON باشد

Console.WriteLine(c) → در خروجی 11 چاپ می‌شود

Console.WriteLine(c.ToString) →

در خروجی Red چاپ می‌شود؛

تبدیل انواع داده ها:

هر وقت لازم باشد که داده های از یک نوع، در متغیرهایی از نوع دیگر قرار گیرند، عمل تبدیل نوع (Type conversion) صورت می گیرد.

دستورات زیر را در نظر بگیرید:

```
Dim a As short = 11
Dim b As long
```

b = a → این نوع تبدیل را تبدیل پهن سازی می گویند که بی خطر است

asb → این نوع تبدیل را تنگ ساز می گویند که داده های با اندازه بزرگتر را به مقادیر کمتر نوع کوچک تبدیل می کند و پرخطر است

1- برای صحیح کردن برنامه نویس به استفاده از توابع تبدیل نوع از

```
option strict ON
```

استفاده کند.

2- توابع تبدیل نوع عبارتند از:

CBool - CByte - Cchar - CDate - CDBL - CInt - Clng
CObj - CShort - CSng - CStr - CType - val - Str

3- تابع CType: CType (مقدار یا کد , نوع مقصد) AS

مقدار در قالب نوع مقصد

مقداری که قرار است تبدیل شود

4.

عملگر:

$a = a + b$

عملوند — عملگر

1- عملگرهای محاسباتی: ^ توان * ضرب + جمع - تفریق / تقسیم اعشاری
\ تقسیم صحیح MOD باقیمانده

2- عملگرهای انتساب: = ^= *= /= \= += -= &=

3- عملگرهای رابطه‌ای: = <> > < >= <= Like

4- عملگرهای اتصال: + &

5- عملگرهای منطقی: NOT And Or AndAlso OrElse

نکته: $a = a + 1 \iff a += 1$

اتصال کوتاه: یکی از فعال‌ترین بخش‌هایی است که توسط کامپایلر به کار می‌رود. در این شیوه بخش‌های زائد شرط که نتیجه‌ی آن‌ها تأثیری بر نتیجه‌ی نهایی ندارد ارزیابی نمی‌شوند و سرعت اجرای دستورات بالا می‌رود. عملگرهای And و OR هر دو عبارات منطقی دو سوی خود را صرف‌نظر از مقدارشان بررسی می‌کنند اما اگر به جای And از AndAlso استفاده شود در صورتی که نتیجه عبارت اول False باشد عبارت دوم ارزیابی نمی‌شود و اگر به جای OR از OrElse استفاده کنید در صورتی که نتیجه عبارت اول True باشد عبارت دوم محاسبه نمی‌شود (ارزیابی نمی‌شود).

عملگر like: نتیجه‌ی آن یا True یا False

```
Dim Str1 As String
Dim Str2 As String
Dim res As Boolean
res = Str1 like Str2
```

نحوه کار با like

1- [ ] : برای یکی از کاراکترها را نشان می‌دهد
2- # : نشان‌دهنده یک کاراکتر عددی است
3- ? : نشان‌دهنده یک کاراکتر است
4- * : نشان‌دهنده تعدادی از کاراکترها است

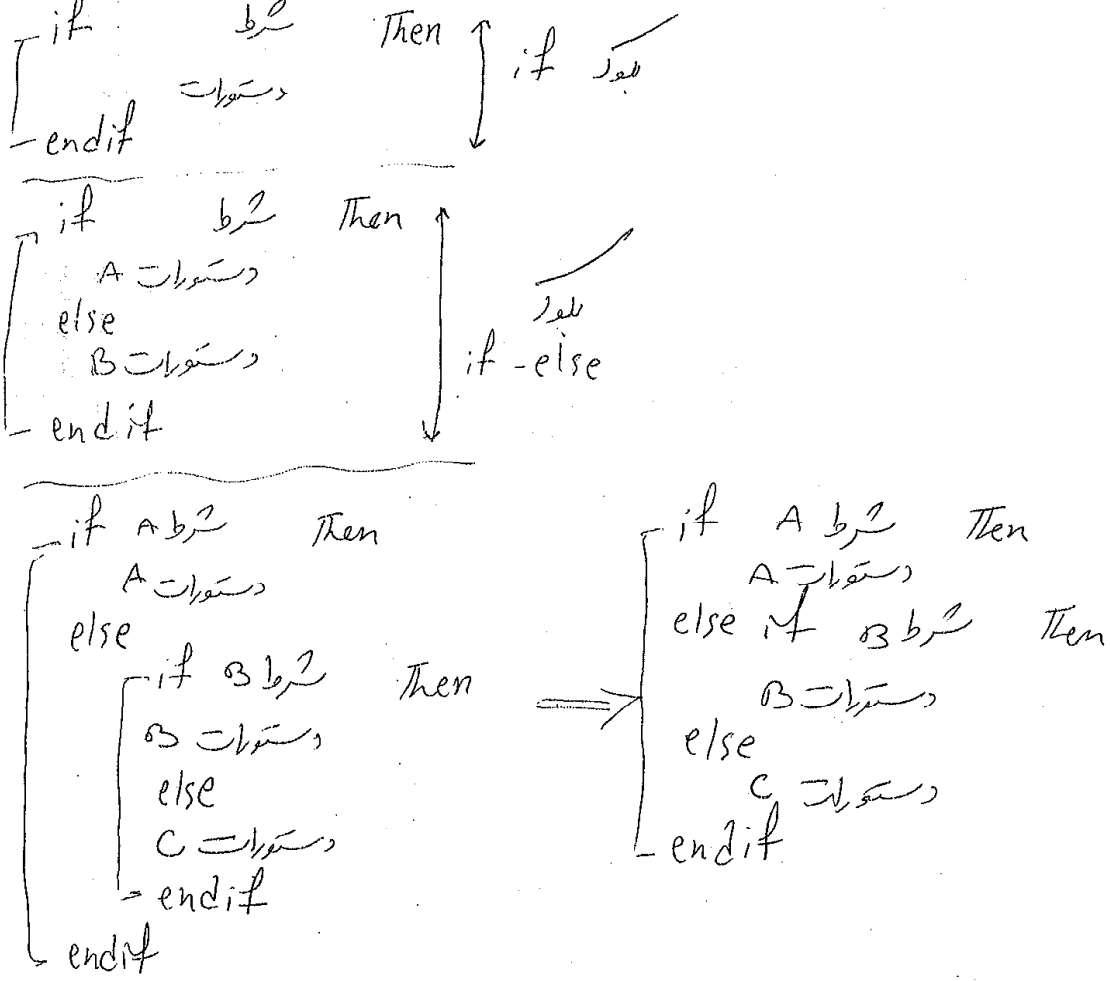
Str1	like	Str2
aBBBa		a*a
F		[A-Z]
F		[!A-Z]
a2a		a#a
BAT123		B?T*
a 45 b		a[L-P]#[!c-e]
.Net		?N*
FRAMEWork		[A-Z]*
CLR 123		?*#2#
1classobject123		1*obj*#23

- برنامه‌ها در VB.Net از دستورات تشکیل می‌شوند. هر دستور یا Statement کاری را که باید توسط کامپیوتر انجام گیرد مشخص می‌کند.
- دستورات مربوط به تصمیم‌گیری در VB.Net به انواع زیر تقسیم می‌شوند:

1- if-then-else:

به کمک دستورات شرطی می‌توان اجرای برنامه را براساس مقدار متغیرها و یا متناسب با آنها به مسیرهای مختلف هدایت کرد.

ساختار این دسته به صورت زیر است:



مثال: توسط ساختار if بررسی کنید که آیا عدد ذخیره شده در متغیر n زوج است یا فرد؟

```
Dim n AS Integer
n = CInt(Console.Readline())
if (n Mod 2 = 0) Then
    Console.writeline("{0} is a even number", n)
else
    console.writeline("{0} is a odd number", n)
endif
```

مثال: ضمن بررسی ساختار if زیر، خروجی نهایی را در صورتی که مقدار متغیر day برابر با 3 باشد، بنویسید؟

```
Dim day AS Integer= 3
if day=1 Then
    Console.writeline("یکشنبه")
else
    if day=2 Then
        Console.writeline("دوشنبه")
    else
        if day=3 Then
            console.writeline("سه شنبه")
        else
            console.writeline("چهارشنبه")
        endif
    endif
endif
```