یادآوری و تکمیل مطالب

یادآوری: یکی از راه‌های ایجاد برنامه به زبان C# (C Sharp) نوشتن برنامه در قالب فایل متنی ساده از طریق ابزاری نظیر **Notepad** است. این پرونده‌ها با پسوند **CS.** ایجاد می‌شود. سپس نوبت ترجمه و تبدیل آن به فایل اجرایی است. برای ترجمه برنامه به شکل زیر از مترجم **CSC** استفاده می‌کنیم:

CSC.EXE نام و مسیر پرونده cs.

با اجرای فرمان فوق در صورتی‌که برنامه اشکال داشته باشد، لیست خطاها ظاهر می‌شود و برنامه‌نویس باید خطاهای به وجود آمده را برطرف نماید و در صورتی‌که خطایی در برنامه نباشد، فایل اجرایی با پسوند **EXE.** ساخته می‌شود. شما کافی است در سطر فرمان نام فایل تولید شده‌ی اجرایی را تایپ کنید و برای اجرا **Enter** را بزنید. این شیوه بسیار وقت‌گیر است و برای عیب‌یابی برنامه‌های بزرگ دشوار می‌باشد.

IDE (محیط برنامه‌نویسی متمرکز): برای نوشتن و عیب‌یابی راحت‌تر برنامه‌ها و تولید فایل اجرایی از **IDE** یا محیط توسعه برنامه متمرکز استفاده می‌کنیم. برخی از **IDE**ها به صورت رایگان و به اصطلاح متن‌باز (**Open Source**) و بعضی دیگر تجاری (**Commercial**) هستند. در سیستم‌عامل ویندوز می‌توان از محیط برنامه‌نویسی رایگان و متن‌باز **SharpDevelop** یا به اختصار **develop#** استفاده نمود. مزیت این **IDE** نسبت به **IDE** مایکروسافت حجم کم آن است. برای استفاده از این **IDE** ابتدا باید نرم‌افزار **Net Framework 4.5.** را نصب کنید سپس نرم‌افزار متن‌باز **develop#** نصب شود. پس از نصب، با کلیک بر روی آیکون **SharpDevelop 5.1** در منوی **Start** برنامه اجرا شده و محیط **IDE** ظاهر می‌شود برای باز کردن یک برنامه و پروژه به شکل زیر عمل می‌کنیم:

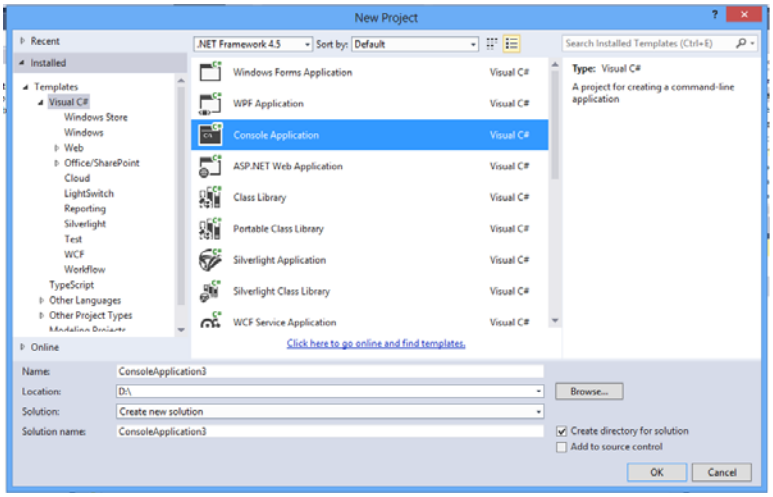
منوی File ← زیرمنوی New ← گزینه Solution

حال در پنجره **New Project** گزینه‌ی **Windows Application** را انتخاب کرده و در قسمت **Name** نامی را برای پروژه انتخاب می‌کنیم و در انتها بر روی دکمه **Create** کلیک می‌کنیم.

■ **نکته‌ی ۱:** در محیط **develop#** از طریق زبانه **design** و **tools** می‌توانیم به فرم اصلی و ابزارهای برنامه‌نویسی دسترسی داشته باشیم.

■ **نکته‌ی ۲:** در محیط **develop#** فرم ابتدایی را **MainForm** می‌نامند.

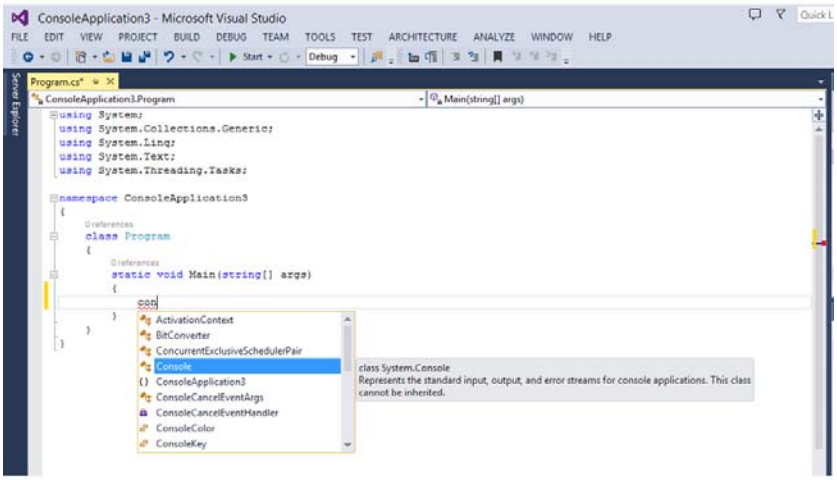
مایکروسافت، **IDE** ویژوال استودیو اکسپرس نسخه ۲۰۱۲ را به‌طور رایگان برای برنامه‌نویسان عرضه کرده است. برای نوشتن برنامه به زبان **C#**، ابتدا در پنجره **Start page** گزینه **New Project** را انتخاب سپس از زیر گزینه‌ی **Visual C#** آن‌گاه **Console Application** را انتخاب می‌کنیم. در ادامه با مشخص کردن نام پروژه و مسیری که در آن پروژه ذخیره می‌شود. بر روی **ok** کلیک می‌کنیم.



و در پنجره ویرایش‌گر برنامه، دستورات را در داخل ( )Main می‌نویسیم.

امکانات IntelliSense: در هنگام تایپ دستورات برنامه، با نوشتن چند حرف از نام دستور (نام کلاس، متد، ...) لیستی از نام متدها و دستورات مرتبط نشان داده می‌شود. کاربر می‌تواند در این لیست با کلیدهای فلش بالا و پایین حرکت کرده تا بر روی کلمه موردنظر قرار گیرد، آن‌گاه از طریق کلیدهای Tab یا فاصله (Space Bar)، دستور موردنظر را برگزیده تا به‌طور خودکار در برنامه قرار بگیرد.

■ **نکته‌ی ۱:** ویژگی IntelliSense، علاوه بر این‌که سبب می‌شود سرعت تایپ برنامه افزایش یابد، باعث می‌شود دستورات با املای صحیح و بدون خطا نوشته شود.



■ **نکته‌ی ۲:** لیست کلمات نمایشی IntelliSense مرتبط با کلمه‌ای است که برنامه‌نویس در حال تایپ کردن آن است.

■ **نکته‌ی ۳:** کلیدهای ترکیبی CTRL + SPACE لیستی مربوط به IntelliSense را در داخل متن برنامه ظاهر می‌کند.

■ **نکته‌ی ۴:** در صورتی که IntelliSense فعال نباشد از طریق زیر آن را فعال کنید.

بخش C# → قسمت Text Editor → گزینه‌ی Options → منوی Tools

گزینه‌ی Auto List members و Parameter information را فعال کنید:

نحوه‌ی ترجمه و اجرای برنامه در VS: برای ترجمه و اجرای برنامه از منوی **Debug** گزینه‌های **Start Debugging** (یا کلید **F5**) و یا **Start Without Debugging** (**Ctrl + F5**) را انتخاب نمایید.

دستورات ورودی و خروجی: برای دریافت اطلاعات از متد ( ) **ReadLine** و یا ( ) **ReadKey** استفاده می‌شود و برای نمایش اطلاعات از متدهای ( ) **WriteLine** و ( ) **Write** استفاده می‌شود.

تبدیل نوع داده: برنامه‌نویس می‌تواند نوع داده‌هایی را که می‌خواهد در متغیر قرار دهد را تغییر و به نوع دیگری تبدیل نماید برای این امر از روش زیر بهره می‌گیریم:

عبارت مبدأ (نوع داده مقصد)

مثال:

```
byte    a=16;
a= (byte) (a+10);
```

ابتدا مقدار متغیر **a** را با عدد **10** جمع می‌کند نتیجه یک مقدار نوع **integer** می‌باشد برای این‌که در متغیر **a** که از نوع **byte** است قرار گیرد، ابتدا تبدیل نوع انجام می‌شود سپس در متغیر قرار می‌گیرد.

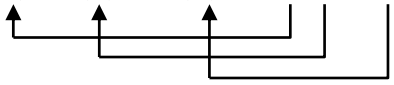
■ **نکته:** در تبدیل نوع داده باید به نوع داده‌های مقصد و مبدأ توجه کرد. نوع داده مقصد باید بتواند مقدار داده مبدأ را در خود جای دهد.

الگوی جای‌گذاری در رشته: در هنگام نمایش اطلاعات در متد ( ) **WriteLine** می‌توان از الگوی جای‌گذاری به شکل زیر استفاده کرد:

System.Console.WriteLine ("الگوی جای‌گذاری" ,1 متغیر ,2 متغیر ، ...) ;

در الگوی جای‌گذاری اولین متغیر به‌جای {۰}، دومین متغیر به‌جای {1}و ... قرار می‌گیرد.

```
int       a = 5 , b = 6;
System. Console. Write Line ("{۰} + {1} = {2}" , a , b , a+b);
```



خروجی: 5+6=11

علاوه بر محل قرارگیری یک متغیر یا عبارت می‌توانیم به شکل زیر الگو و طریقه‌ی نمایش و نیز تراز چپ و راست را مشخص کرد.

{الگوی نمایش: عدد تراز , شماره}

- عدد تراز به تعداد فضاهای خالی که جهت نمایش مقدار متغیر یا عبارت اختصاص داده می‌شود اطلاق می‌گردد. اگر عدد تراز منفی باشد مقدار موردنظر در فضای اختصاص یافته چپ‌چین می‌شود و اگر عدد تراز مثبت باشد، مقدار موردنظر در فضای اختصاص یافته راست‌چین می‌شود.

■ **نکته:** هرگاه عدد تراز اعلام شده به اندازه تعداد ارقام عدد نمایشی نباشد عدد تراز در نمایش ارقام عدد تاثیر ندارد و عدد عیناً نمایش می‌یابد ولی اگر عدد تراز بیش‌تر از تعداد ارقام عدد باشد اگر عدد تراز مثبت باشد چون تراز راست چین است پس به تعداد ارقام اضافه‌تر در تراز فضای خالی در سمت چپ عدد نمایش ظاهر می‌شود و اگر عدد تراز منفی باشد چون تراز چپ چین است پس به تعداد ارقام اضافه‌تر در تراز فضای خالی در سمت راست عدد نمایشی ظاهر می‌شود.

مثال:

Console.WriteLine ("Sample:{0,5}",36); ⟶ (خروجی) **Sample:** □□□35

۵ فضای راست چین

Console.WriteLine ("Sample:{0,5}",364); ⟶ (خروجی) **Sample:** □□364

۵ فضای راست چین

Console.WriteLine ("Sample:{0,-5}",36); ⟶ (خروجی) **Sample:** 35□□□

۵ فضای چپ چین

Console.WriteLine ("Sample:{0,-5}",364); ⟶ (خروجی) **Sample:** 364□□

۵ فضای چپ چین

- الگوی نمایش از تعدادی کاراکتر تشکیل شده که نحوه‌ی نمایش داده را معین می‌کند.

Specifier	Format
C	مقدار پولی خاص یک محل
D	انواع صحیح (integer)
E	نماد علمی
F	نقطه اعشار ثابت
G	اعداد عمومی
N	نقطه اعشار ثابت با جدا کننده کاما
P	اعداد دارای درصد
X	هگزادسیمال

کاراکتر	معنی
#	عدد
.	اعشار
,	جدا کننده سه رقمی
%	درصد
0	اضافه کردن صفر
;	اعداد مثبت، منفی و مقادیر صفر را با قالب‌های خاصی نمایش می‌دهد.
E0 E+0 E – 0 e0 e+0 e – 0	نماد علمی

■ **نکته:** هرگاه الگوی نمایش F باشد عدد با دو رقم اعشار نمایش می‌یابد اگر بخش اعشار عدد بیش از دو رقم باشد فقط دو رقم اول بخش اعشار به صورت گرد شده نمایش می‌یابد در هنگام نمایش به رقم سوم اعشار توجه می‌کند اگر پنج و یا فراتر از ۵ باشد یک واحد به بخش اعشار نمایش اضافه می‌کند ولی اگر کم‌تر باشد بخش اعشار دو رقم نمایش‌اش تغییر نمی‌یابد.

مثال:

Console.Write(“{0,1:F}{1,1:F}{2,1:F}”,23,23.754,23.755);

خروجی به ترتیب 23.00 , 23.75 , 2376 می‌باشد.

■ **نکته۲:** هرگاه الگوی نمایش N باشد بخش صحیح عدد از سمت راست سه رقم سه رقم با کاما جدا شده نمایش می‌یابد و همانند الگوی نمایشی F بخش اعشار با دو رقم اعشار نمایش می‌یابد.

مثال:

```
Console.Write("{0,1:N}{1,1:N}",7564.358,7564.352};
```

خروجی به ترتیب 7564.36 و 7564.35 خواهد بود.

دستور if- else: این دستور برای تصمیم‌گیری در اجرای دستورات استفاده می‌شود. در جلوی دستور **if** یک عبارت منطقی ساده و یا مرکب به عنوان شرط نوشته می‌شود هرگاه شرط **true** باشد دستور یا دستورات بعد از این عبارت اجرا می‌شود و اگر نتیجه شرط **false** باشد دستور یا دستورات بعد از **else** اجرا می‌شود.

■ **نکته:** در زبان‌های برنامه‌نویسی، امکاناتی وجود دارد که نیاز به حفظ کردن تمام جزئیات و تایپ دستورات نیست، برای نمونه با کلیک راست کردن در پنجره‌ی کدنویسی، منویی ظاهر می‌شود بر روی گزینه **Insert Snippet** کلیک نمایید در این صورت منوی دیگری باز می‌شود از آن گزینه **Visual C#** را انتخاب کنید در این صورت لیستی از کلمات رزرو شده سی شارپ را نمایش می‌دهد دستور موردنظر را از این لیست یافته و انتخاب کنید تا دستور در بخش برنامه نویسی افزوده شود.

متد ( ) Parse: متدی است که رشته‌ی عددی را به عدد تبدیل می‌کند.

عدد ← (رشته عددی) Parse. نوع داده

مثال:

```
int     x = int.Parse("375"); //x = 375
float   y = float.Parse("57.62"); //y = 57.62
```

■ **نکته:** اگر کاربر در هنگام ورود داده‌های عددی کاراکتری غیر از عدد وارد کند، در هنگام تبدیل پیام خطا ظاهر می‌شود.

مثال:

```
int     a = int.Parse("37m5");
```

خطا ظاهر می‌شود چون **m** قابل تبدیل به عدد نمی‌باشد.

متد ( ) TryParse: این متد همانند متد **( ) Parse** رشته را به عدد تبدیل می‌کند شکل کلی این متد به صورت زیر است:

مقدار نوع bool ← (متغیر عددی مقصد و متغیر رشته عددی مبدأ) TryParse. نوع داده

هرگاه مقدار متغیر رشته عددی فقط شامل کاراکترهای عددی باشد، این تبدیل بدون مشکل انجام شده و مقدار عددی در متغیر مقصد ذخیره و خروجی متد **TryParse** مقدار **true** می‌شود و اگر در رشته عددی مقداری غیرعددی باشد تبدیل نمی‌تواند انجام شود و مقداری در متغیر عددی مقصد قرار نمی‌گیرد و خروجی متد برابر با **false** می‌شود.

مثال:

```
int x; bool y; string z;
z = "35";
y = int.TryParse(z,x);     // x = 35    ,    y = true
z = "35k";
y = int.TryParse(z,x);     // x = 0     ,    y = false
```

عملگر سه‌تایی یا علامت سوال: این عملگر خلاصه شده‌ی فرمان **if** می‌باشد که شکل کلی آن به صورت زیر است:

(عبارت منطقی) ? مقدار اول : مقدار دوم ;

در این عملگر ابتدا عبارت منطقی محاسبه می‌شود اگر حاصل این عبارت **true** شود مقدار اول و اگر **false** شود مقدار دوم باز می‌گردد.

مثال: برنامه زیر دو عدد از ورودی دریافت و بزرگترین عدد را نمایش می‌دهد.

```
System.Console.Write("Number 1 : ");
int   a = int.Parse(System.Console.ReadLine());
System.Console.Write("Number 2 : ");
int   b = int.Parse(System.Console.ReadLine());
int   max = (a > b) ? a : b;
System.Console.WriteLine(max);
```

دستور switch: این دستور برای بررسی حالت‌های مختلف یک عبارت به کار می‌رود.

دستورات تکرار یا حلقه: هرگاه بخواهیم یک یا چند دستور را بیش از یکبار انجام دهیم از حلقه‌ها استفاده می‌کنیم.

دستور while: دستور یا دستورات را تا زمانی که حاصل عبارت منطقی **while** برابر **true** باشد اجرا می‌شود.

مثال: برنامه‌ی زیر اعداد یک تا نه را نمایش می‌دهد.

```
int       a = 1;
while     (a <= 9)
System.Console.WriteLine(a++);
```

در برنامه‌ی فوق ابتدا مقدار **a** نمایش می‌یابد سپس به مقدار **a** یک واحد اضافه می‌کند.

دستور for: دستور یا دستورات را به تعداد معین تکرار و اجرا می‌کند.

مثال: برنامه‌ی زیر اعداد فرد دو رقمی را نمایش می‌دهد.

```
for (int i=11; i<=99; i+=2)
System. Console.WriteLine(i);
```

ثابت‌ها (Constant): ثابت‌ها شناسه‌هایی هستند که در طول اجرای برنامه مقدارشان تغییر نمی‌یابد. شکل کلی معرفی ثابت‌ها به صورت زیر است:

const نوع داده ثابت نام = مقدار ;

مثال:

```
const int m=50;
```

حلقه‌های متداخل (تودرتو): هرگاه یک دستور تکرار را در داخل دستور تکرار دیگری به کار ببریم حلقه تودرتو یا متداخل نامیده می‌شوند.

مثال: برنامه‌ی زیر کاراکتر * را در پنج سطر و در هر سطر سه بار به صورت زیر نشان می‌دهد.

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

۵ سطر

```
for(int i = 1;i <= 5;++i)
{
        for(int  j = 1; j <= 3;++j)
                System.Console.Write("*");
        System.Console.WriteLine( );
}
```

کاراکتر \ : هرگاه کاراکتر \ در رشته‌ای دیده شود، کاراکتر بعدی آن اثر و عملکرد خاصی دارد. به عبارت دیگر کاراکتر \ ، سبب تغییر عملکرد کاراکتر بعد از خودش می‌شود. به همین دلیل به این مجموعه کاراکترها که کاراکتر اول \ و دومی کاراکتر دیگری است دنباله فرار (Escape Sequence) گفته می‌شود.

عملکرد	دنباله
ایجاد یک بوق هشدار[۲]	\a
حذف یک کاراکتر(Backspace)	\b
ایجاد یک خط خالی(New Line)	\n
ایجاد یک فاصله افقی زیاد tab	\t
ایجاد یک تک کوتیشن (')	\'
ایجاد یک دابل کوتیشن (")	\"
ایجاد یک Back slash (\)	\\

■ **نکته:** زبان C# روی حروف کوچک و بزرگ حساس است و این شامل دنباله‌های جدول بالا نیز می‌شود.

مثال:

```
Console.WriteLine("Hello \t student \t welcome);
```

خروجی به صورت زیر است:

```
Hello    student
welcome
```

پرسش‌های چهار گزینه‌ای

۱. برای نوشتن یک برنامه ساده به زبان C# از کدام گزینه می‌توان برای نوشتن استفاده کرد؟

۱) Wordpad , Notepad
۲) Notepad+ , Notepad
۳) Wordpad , Notepad+
۴) Wordpad , Notepad+ , Notepad

۲. دستور ترجمه‌ی برنامه‌ای به‌نام P1 چیست؟

۱) CSC.EXE P1.TXT
۲) P1.TXT CSC.EXE
۳) CSC.EXE P1.CS
۴) P1.CS CSC.EXE

۳. با کامپایل برنامه‌ای به‌نام M1 چه فایلی به وجود می‌آید؟

۱) M1.CS
۲) M1.com
۳) M1.C#
۴) M1.EXE

۴. تمام گزینه‌های زیر جزء ویژگی‌های IDE می‌باشد به جز:

۱) محیط برنامه‌نویسی متمرکز می‌باشد.
۲) نوشتن برنامه راحت‌تر است.
۳) اجرای برنامه راحت‌تر انجام می‌شود.
۴) عیب یابی و خطایابی در آن راحت‌تر است.

۵. تمام گزینه‌های زیر درباره IDE، develop# صحیح است، به جز:

۱) open source می‌باشد.
۲) تحت سیستم‌عامل ویندوز اجرا می‌شود.
۳) نسبت به IDE مایکروسافت کم‌حجم‌تر است.
۴) commercial می‌باشد.

۶. در محیط develop# فرم ابتدایی چه نام دارد؟

۱) MainForm ۲) ParentForm ۳) IDE Form ۴) MDI Form

۷. به لیستی که در هنگام تایپ دستورات، با نوشتن چند حرف از نام متد ظاهر می‌شود، چه می‌گویند؟

۱) List Method ۲) IntelliSense ۳) Debugging ۴) Tools

۸. برای ظاهر کردن لیست IntelliSense در داخل متن برنامه از چه کلید یا کلیدهایی استفاده می‌کنیم؟

۱) Tab ۲) space bar ۳) Enter ۴) Ctrl + space

۹. برای ترجمه و اجرای برنامه از کدام منو استفاده می‌شود؟

۱) Run ۲) File ۳) Debug ۴) Project

۱۰. کدام گزینه برای اجرا و ترجمه‌ی برنامه بدون بررسی اشکالات برنامه به‌کار می‌رود؟

۱) F5 ۲) Ctrl+F5 ۳) F6 ۴) Ctrl+F6

۱۱. نتیجه‌ی دستور مقابل چیست؟

```
byte    a=100;
a=a+150;
```

۱) در a مقدار ۲۵۰ قرار می‌گیرد.

۲) خطا می‌دهد. چون عدد ۲۵۰ در محدوده‌ی متغیر a و نوع byte قرار ندارد.

۳) خطا می‌دهد. چون نوع داده انتسابی با نوع متغیر a سازگار نمی‌باشد.

۴) خطا می‌دهد. زیرا بجای علامت == از = استفاده گردیده است.

۱۲. با توجه به تعریف مقابل کدام انتساب زیر به‌درستی انجام می‌گیرد؟

```
byte  a=10; int   b=8;
```

الف) b=a+b; ب) a=a+b;

ج) a=(byte)a+b د) a=a+4;

۱) الف، د ۲) الف، ج ۳) ب و ج، د ۴) الف، ب، د

۱۳. خروجی دستورات زیر چیست؟

```
int  a=2 , b=9;
System.Console.WriteLine("{1}+{0}={2}" , a+b , b , a);
```

۱) 2+9=11 ۲) 9+2=11 ۳) 2+11=9 ۴) 9+11=2

۱۴. نوع داده‌ی short به کدام نوع به‌طور ضمنی تبدیل نمی‌شود؟

۱) int ۲) byte ۳) long ۴) float

۱۵. با توجه به تعریف‌های مقابل کدام‌یک از انتساب‌های زیر صحیح نمی‌باشد؟

```
int a=4; short b; byte c; sbyte d; long e;
```

۱) b=a; ۲) c=a; ۳) d=a; ۴) e=a;

۱۶. تمام گزینه‌های زیر می‌توانند به‌طور ضمنی به نوع **int** تبدیل شوند به جز:

۱) sbyte ۲) long ۳) ushort ۴) byte

۱۷. نتیجه‌ی دستورات مقابل چیست؟

```
int   a=(int) 19.99;
System.Console.WriteLine (a);
```

۱) 19 ۲) B ۳) A ۴) 65

۱۸. نتیجه دستورات زیر چیست؟

```
char   m=(char) (byte)((char) 65+(char)3);
System.Console.WriteLine(m);
```

۱) 68 ۲) A ۳) D ۴) E

۱۹. خروجی دستورات زیر چیست؟

```
int   m=20 ,  n=20;
System . Console . WriteLine ("m={0,10}",m);
System . Console . WriteLine ("n={0,-10}",n);
```

۱) m = 20
n = 20

۲) m = 20
n = 20

۳) m = 20
n = 20

۴) m = 20
n = 20

۲۰. با اجرای فرامین زیر بین دو عدد 5 و 6 چند فضای خالی درج می‌گردد؟

```
int     a=5 ,  b=6;
System . Console . WriteLine ("{0, -3} {1,2}",a,b);
```

۱) 1 ۲) 3 ۳) 5 ۴) 6

۲۱. با اجرای فرامین زیر بین دو عدد 5 و 6 چند فضای خالی درج می‌شود؟

```
int     a=5 ,  b=6;
System . Console . WriteLine ("{0, 3} {1,-2}",a,b);
```

۱) 3 ۲) 2 ۳) 5 ۴) 0

۲۲. با اجرای فرامین زیر عددهای ۵۵ و ۷ و ۴۴ در چه ستون‌هایی نمایش می‌یابد؟ (به ترتیب از راست به چپ)

```
int     a= 55 , b=7 , c=44;
System . Console . WriteLine ("{0,5} {1,-3} {2,4}",a,b,c);
```

۱) ۵، ۷، ۱۲ ۲) ۴، ۶، ۱۲ ۳) ۴، ۶، ۱۱ ۴) ۱، ۸، ۹

۲۳. خروجی کدام گزینه با سایر گزینه‌ها متفاوت است؟

```
int     a=57;
```

۱) `System . Console . WriteLine ("{0,0}" ,a);`
۲) `System . Console . WriteLine ("{0,2}" ,a);`
۳) `System . Console . WriteLine ("{0,-3}" ,a);`
۴) `System . Console . WriteLine ("{0,3}" ,a);`

۲۴. با اجرای فرمان های زیر رشته ALI در کدام ستون نمایش می‌یابد؟

```
int       a=222 , b=11 ; string c="ALI";
System . Console . WriteLine ("{0,1},{1,6} {2,5}",a,b,c);
```

۱) 14 ۲) 13 ۳) 12 ۴) 11

۲۵. متد ( ) Parse چه وظیفه‌ای را برعهده دارد؟

۱) تبدیل عدد به رشته
۲) تبدیل مقادیر منطقی به عدد
۳) تبدیل رشته‌ی عددی به عدد
۴) تبدیل کاراکتر به عدد

۲۶. کدام دستور با خطا مواجه نمی‌شود؟

۱) `int b = int.Parse("30/2");`
۲) `int a = int.Parse("25k32");`
۳) `int c = int.Parse("5"+"2");`
۴) `int d = int.Parse("78+2");`

۲۷. نتیجه اجرای دستورات زیر چیست؟

```
int x; bool y; string z = "91";
y = TryParse(z, x);
```

۱) x = 91 , y = true
۲) x = 91 , y = false
۳) x = 9 , y = true
۴) x = 9 , y = false

۲۸. نتیجه اجرای دستورات زیر چیست؟

```
int x; bool y; string z = "75m3";
    y = TryParse(z, x);
```

۱) x = 35, y = true
۲) x = 35, y = false
۳) x = 0, y = true
۴) x = 0, y = false

۲۹. عملکرد برنامه زیر چیست؟

```
int x = int.Parse(System.Console.ReadLine( ));
x = (x > 0)? x : -x;
System.Console.WriteLine(x);
```

۱) منفی عدد ورودی را نمایش می‌دهد.
۲) خود عدد ورودی را نمایش می‌دهد.
۳) قرینه‌ی عدد ورودی را نمایش می‌دهد.
۴) قدرمطلق عدد ورودی را نمایش می‌دهد.

۳۰. خروجی برنامه‌ی زیر چیست؟

```
int x = 6%4, y = 4%6;
x = (x > y)? x + y : y * x;
y = (x > y)? x % y : y % x;
System.Console.WriteLine("{0},{1}", x, y);
```

۱) 8,0
۲) 8,2
۳) 6,2
۴) 6,1

۳۱. خروجی برنامه‌ی زیر چیست؟

```
int   a = 3;
while (a < 12)
      a = (a%2 == 0)? a + 3 : a + 5;
System.Console.Write(a);
```

۱) 14
۲) 16
۳) 3 8 11
۴) 3 8 11 16

۳۲. کدام قالب‌بندی برای اعداد به‌صورت نماد علمی به‌کار می‌رود؟

۱) C
۲) E
۳) F
۴) G

۳۳. خروجی دستور زیر چیست؟

```
System.Console.WriteLine("{0,3:e}",574683);
```

۱) 5.24 ۲) 5.746830e+005 ۳) 5.75e+005 ۴) 5.75

۳۴. خروجی دستورات زیر چیست؟

```
System.Console.WriteLine("{0,5:e}",765348251);
System.Console.WriteLine("{0,5:e}",78154682);
```

۱) 7.653483e+008
7.815468e+007

۲) 7.653482e+008
7.815468e+007

۳) 7.653483e+008
7.815469e+007

۴) 7.65348e+008
7.8154869e+007

۳۵. کدام قالب اعداد را با نقطه اعشار ثابت و با جداکننده‌ی کاما نمایش می‌دهد؟

۱) G ۲) P ۳) X ۴) N

۳۶. خروجی دستور زیر چیست؟ (□ به معنی فضای خالی است)

```
System.Console.WriteLine("{0,3:X}",107);
```

۱) 6B □ ۲) 107 ۳) □ 6B ۴) 107 □

۳۷. کدام‌یک از اعلان‌های زیر صحیح است؟

۱) const a = 20 int;
۲) const int a = 20;
۳) const int a = 208;
۴) const a = 208 int;

۳۸. خروجی برنامه زیر چیست؟

```
int i;
for (i = 3; i <= 20; i = i + 5);
System.Console.Write(i);
```

۱) 23 18 13 8 3 ۲) 3 ۳) 23 ۴) خطا ظاهر می‌شود.

۳۹. با اجرای دستورات زیر، چند بار کلمه‌ی ok نمایش می‌یابد؟

```
for (int i = 5; i >= –5; i– – = 2)
for (int j = 1; j <= 15; j+ = 3)
        System.Console.WriteLine("ok");
```

۱) 30 ۲) 11 ۳) 25 ۴) 10

۲ آرایه

آرایه: آرایه، مجموعه‌ای از چندین خانه‌ی متوالی است که برای نگه‌داری عناصر هم‌نوع درحافظه‌ی اصلی و تحت یک نام تعریف می‌گردد. به هر یک از عناصر آرایه یک عنصر یا عضو آرایه اطلاق می‌گردد. برای متمایز کردن خانه‌ها از هم‌دیگر و برای تفکیک و دسترسی به مکان‌های آرایه از یک عدد صحیح به نام اندیس (**index**) استفاده می‌شود.

دلیل استفاده از آرایه: گاهی اوقات در برنامه‌ها نیاز داریم تا تعدادی متغیر همنوع تعریف نماییم. اگر تعداد آن‌ها زیاد باشد، تعریف تک‌تک آن‌ها آسان نمی‌باشد ولی به کمک آرایه یک دفعه گروهی از متغیرها هم‌نام و هم‌نوع را یک‌جا تعریف خواهیم کرد.

تعریف یا اعلان آرایه: در زبان C# برای تعریف یک آرایه باید دو مرحله‌ای عمل کنیم:

الف- اعلان آرایه ب- تخصیص حافظه به آرایه

در اعلان آرایه، نوع عناصر آن مشخص می‌شود، ولی در تخصیص حافظه به آرایه، تعداد عناصر آرایه مشخص می‌گردد. این دو گام را می‌توان در یک دستور یا دو دستور مجزا نوشت.

گام اول (اعلان آرایه):

نوع داده [ ] نام متغیر آرایه ;

در گام اول هنوز مکانی از حافظه برای آرایه تخصیص نیافته و فقط نوع آرایه اعلام گردیده است.

گام دوم (تخصیص حافظه به آرایه):

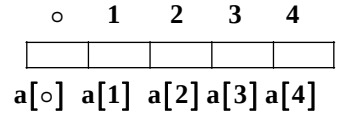
نام متغیر آرایه =new نوع داده [اندازه آرایه] ;

با انجام گام دوم تعدادی خانه‌ی حافظه به آرایه تخصیص یافته است.

مثال:

```
int[ ]  a;
a = new  int[5];
```

با اجرای فرامین فوق آرایه‌ای به شکل زیر بوجود می‌آید.



به یاد داشته باشید همیشه اولین عنصر آرایه دارای اندیس صفر می‌باشد و برای خانه‌های بعدی این اندیس‌ها به ترتیب ۱، ۲، ۳ و ... خواهد بود.

نمونه‌ی اعلان آرایه با یک دستور:

نوع داده [ ] نام آرایه =new نوع داده [اندازه آرایه] ;

مثال:

```
int[ ]  a = new int[5];
```

■ **نکته‌ی ۱:** در هنگام اعلان آرایه، اندازه‌ی آرایه نمی‌تواند عدد منفی باشد. در این صورت پیام خطا ظاهر می‌شود.

■ **نکته‌ی ۲:** هرگاه اندازه‌ی آرایه را عدد صفر اعلام کنید، آرایه فقط یک عنصر خواهد داشت.

مثال:

```
                    ○
int[ ]  x = new int[○]   x
                         x[○]
```

■ **نکته‌ی ۳:** اندازه‌ی آرایه را نمی‌توان یک عدد اعشاری اعلام کرد در این صورت پیام خطا ظاهر می‌شود.

مثال:

```
int[ ]  a = new  int[3.8];
```

■ **نکته‌ی ۴:** اگر اندازه‌ی آرایه یک عدد اعشاری باشد، ابتدا می‌بایست از طریق تبدیل نوع، به نوع عددی صحیح تبدیل شود و سپس در تعریف آرایه به‌کار رود.

مثال:

```
int[ ] y = new   int[(int)3.8];
```

ابتدا عدد اعشاری **3.8** از طریق نوع **int** به عدد صحیح ۳ تبدیل شده و آرایه‌ای به نام **y** با طول ۳ معرفی گردید.

■ **نکته‌ی ۵:** اندازه‌ی آرایه را می‌توان از طریق متغیر نوع صحیح نیز اعلام کرد.

مثال:

```
int      x = 3;                  ○  1  2
float[ ] y = new float[x];       x
```

■ **نکته‌ی ۶:** فضایی که هر آرایه در حافظه‌ی اصلی به خود تخصیص می‌دهد، از رابطه‌ی زیر بدست می‌آید:

فضای نوع آرایه * تعداد خانه‌های آرایه= فضای آرایه

```
int[ ]x = new  int[5];
double[ ]y = new double[3];
```

x فضای آرایه = ۵ (تعداد خانه‌های آرایه) *۴ (فضای نوع **int**) = ۲۰**byte**

y فضای آرایه = ۳ (تعداد خانه‌های آرایه) *۸ (فضای نوع **double**) = ۲۴ **byte**

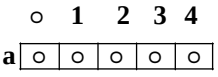
نحوه‌ی دسترسی به خانه‌های آرایه:

برای آن‌که به خانه‌های آرایه دسترسی پیدا کنیم نام آرایه را به همراه اندیس آن به شکل زیر به کار می‌بریم:

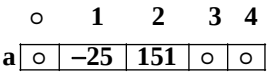
نام متغیر آرایه[اندیس]

مثال:

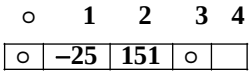
```
int[ ]a = new  int[5];
```




```
a[1] = –25;
a[2] = 151;
a[4] = int.Parse(System.Console.ReadLine());
```

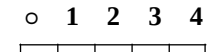


عددی از ورودی خوانده شده و به صورت عددی صحیح در خانه با اندیس ۴ قرار می‌گیرد.



■ **نکته‌ی ۱:** هرگاه بخواهیم به خانه‌هایی از آرایه دسترسی داشته باشیم که آن عنصر وجود نداشته باشد پیام خطا ظاهر می‌شود.

```
int[ ]  a = new int[5];
```




```
a[5] = 10;        خطا می‌دهد
a[–1] = 15;       خطا می‌دهد
```

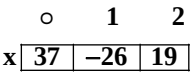
■ **نکته‌ی ۲:** آرایه‌ها در زمان تعریف همانند متغیرها دارای مقدار اولیه می‌باشند. مثلاً اگر آرایه از نوع عددی باشد مقدار صفر و اگر از نوع **bool** باشد دارای مقدار **false** خواهد بود.

■ **نکته‌ی ۳:** کاربر می‌تواند در هنگام تعریف آرایه، مقادیری را به عناصر آن اختصاص دهد. برای این منظور از علامت‌های { } برای معرفی مقادیر عناصر آرایه استفاده می‌شود.

```
نوع داده [ ] نام آرایه =new نوع داده [اندازه آرایه] {مقدار ۱ ، مقدار ۲ ، ... };
```

مثال:

```
int[ ] a = new  int[3] {37,–26,19};
```



■ **نکته‌ی ۴:** در هنگام مقداردهی اولیه به آرایه، تعداد مقادیر باید دقیقاً با تعداد عناصر آرایه برابر باشد.

مثال:

```
int[ ]  x = new  int[3]  {7,8,9,11};      خطا می‌دهد
int[ ]  y = new  int[5]  {3,6,8};         خطا می‌دهد
```

- تعداد مقادیر برای دستور اول بیشتر از تعداد عناصر آرایه می‌باشد.
- تعداد مقادیر برای دستور دوم کمتر از تعداد عناصر آرایه می‌باشد.

■ **نکته‌ی ۵:** در هنگام اعلان و تعریف آرایه می‌توانیم تعداد عناصر آرایه (اندازه یا طول آرایه) را از طریق مقادیر مشخص کنیم با این شیوه هم طول آرایه تعیین می‌شود و هم آرایه دارای مقدار اولیه خواهد بود.

```
(۱)  نوع داده [ ] نام آرایه =new نوع داده [ ] {مقدار ۱ ، مقدار ۲ ، .... مقدار n};
```

یا

```
(۲)  نوع داده [ ] نام آرایه ={مقدار ۱ ، مقدار ۲ ، .... مقدار n};
```