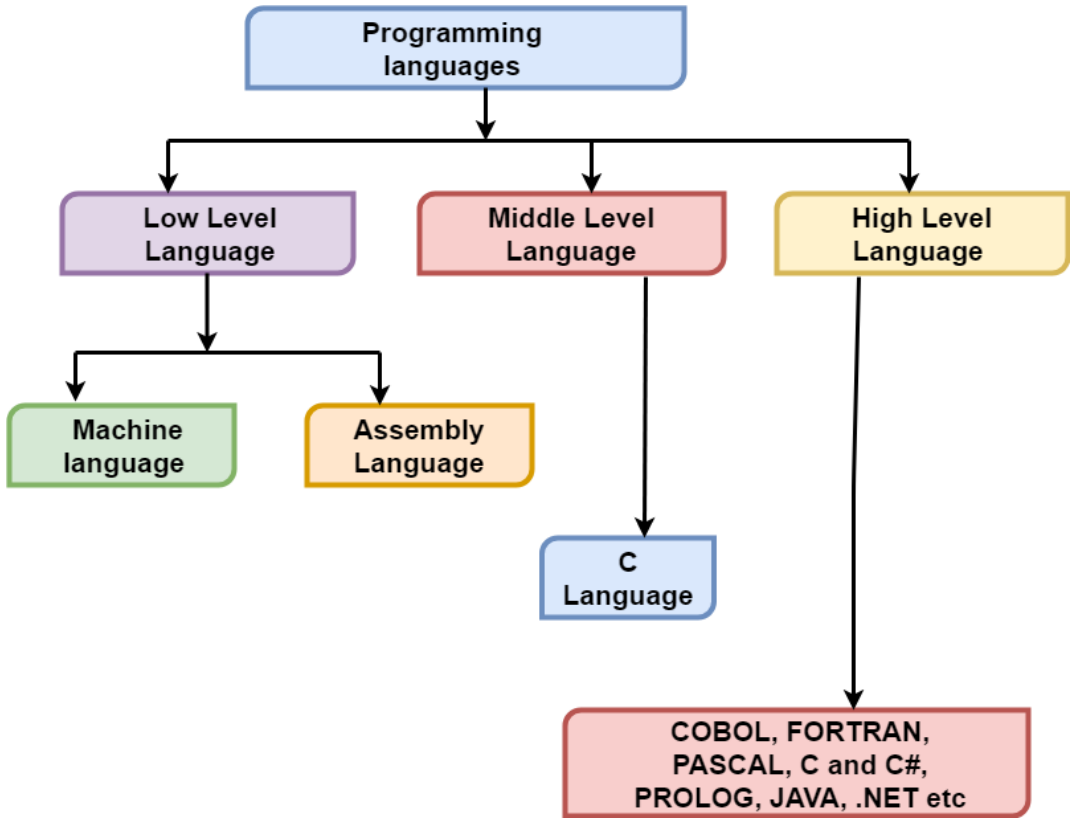
بخش اول

معرفی زبان C

تاریخچه زبان C

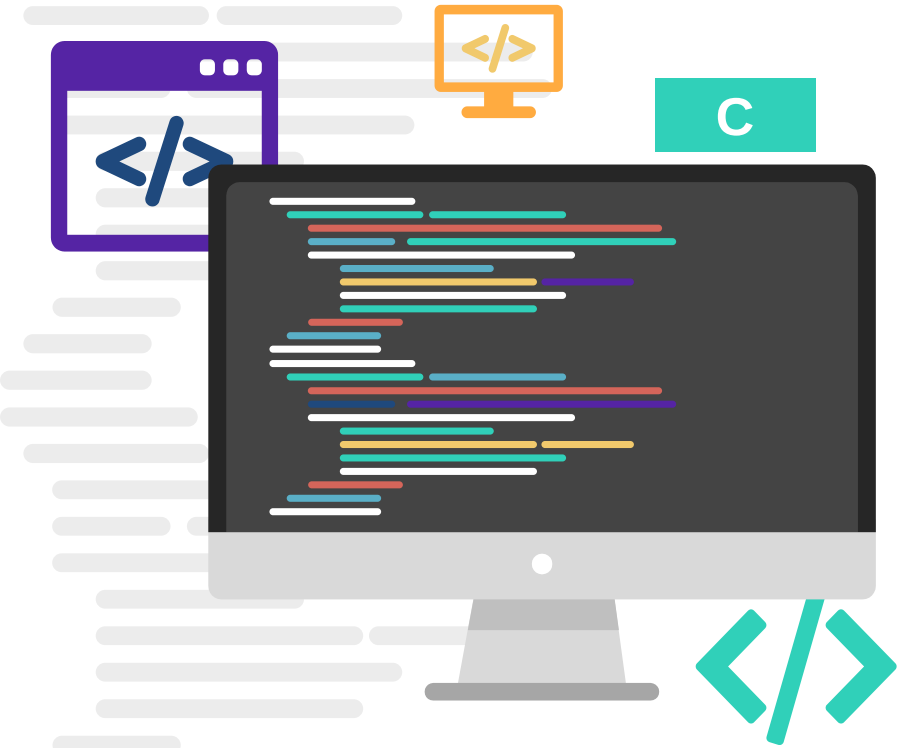
- درسال 1969 تا 1973زبان C توسط دنیس ریچی در آزمایشگاه Bell طراحی شد.

انواع زبان برنامه نویسی



- زبان های برنامه نویسی به سه دسته تقسیم میشوند:
- 1- زبان سطح بالا: نزدیک به زبان انسان، قابلیت خوانایی بالایی دارند مانند زبان python
- 2- زبان سطح میانی: تلفیقی از زبان سطح بالا و سطح پایین مانند زبان C
- 3- زبان سطح پایین: نزدیک به زبان ماشین، دسترسی به بیت و بایت و آدرس مانند زبان Assembly

چرا زبان C



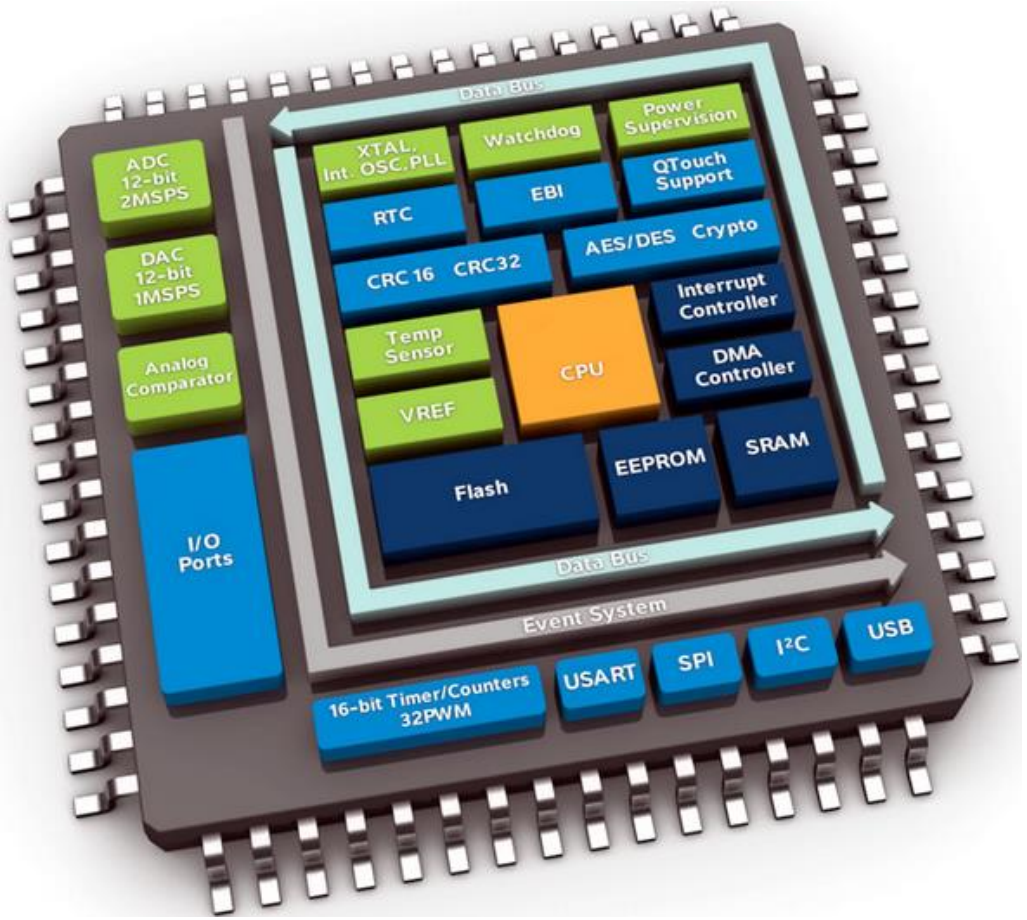
- 1-برنامه نویسیModular
- 2- بازدهی بالا
- 3- وجود استاندارد های مختلف
- 4- وجود سیستم عامل بی درنگ Real-timeoperatingsystem

Embedded System

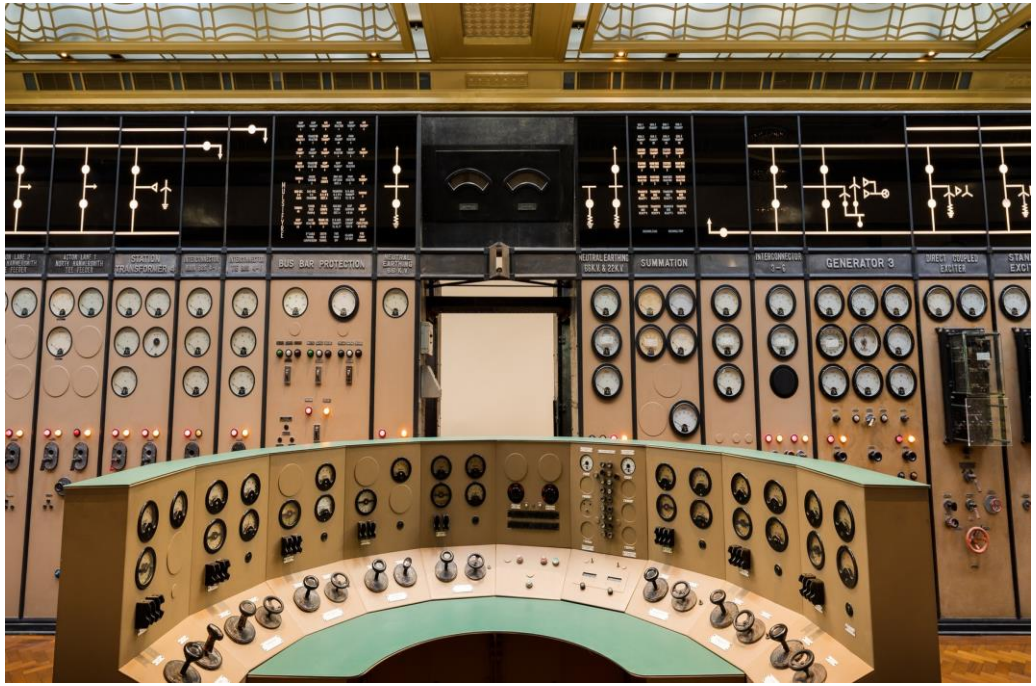
پردازنده ای که داخل محصولات قرار میگیرد را Embedded میگویند، به گونه ای که از دید دیگران پنهان شده است و کاربران معمولا از وجود آن آگاه نیستند، این پردازنده می تواند میکروکنترلر باشد.

میکروکنترلر چیست

- یک رایانه کوچک در ابعاد یک تراشه، که همچون یک ماشین منظم و دیجیتال است و همانند سایر کامپیوتر ها برنامه ای که بر روی آن قرار داده شده است را اجرا می کند.

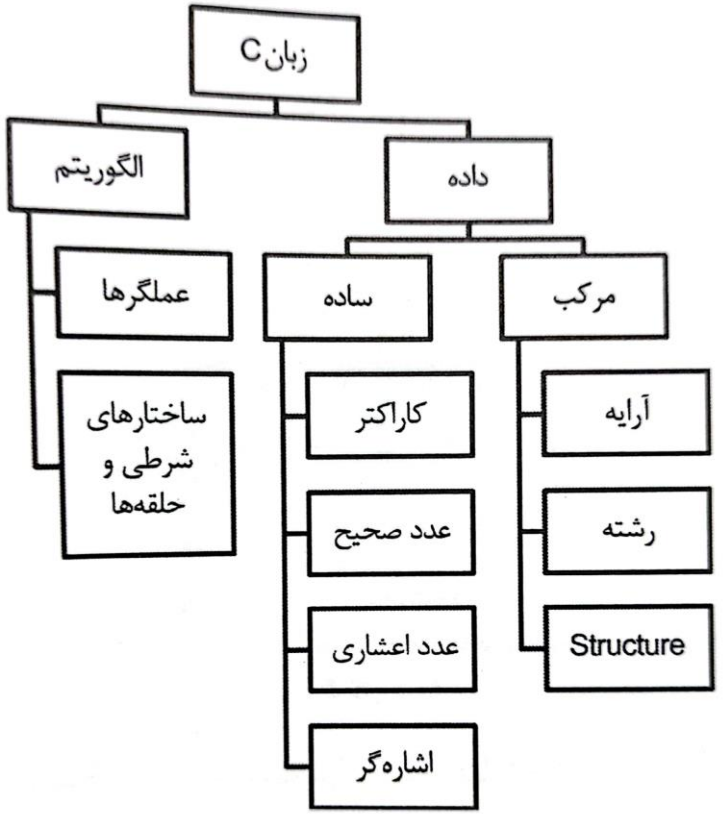


ثبات یا رجیستر



- واحد های مختلف میکروکنترلر توسط اتاق کنترل، کنترل می شوند. هر واحد در این اتاق دارای کلید های مختلفی است که این کلید ها برای راه اندازی و تنظیمات هر بخش است، به این کلیدها ثبات یا رجیستر گفته میشود.

مفاهیم اولیه زبانC



- مواد اولیه در برنامه نویسی عبارتند از داده ها و الگوریتم ها

برنامه<== الگوریتم + داده

مفاهیم اولیه زبانC

- زبان C به حروف بزرگ و کوچک حساس است و تمام کلمات کلیدی ای زبان با حروف کوچک نوشته میشوند.
- هر خط دستور به ; خطم میشود
- حداکثر طول یک خط دستور 255 کاراکتر می باشد، یک خط دستور میتواند در یک یا چند سطر ادامه داشته باشد و یا چندین خط دستور را در کنار هم تایپ کرد.
- برا ایجاد یک خط کامنت از // در ابتدای خط استفاده می شود و برای ایجاد چند خط کامنت از /* و */ در ابتدا و انتهای خطوط کامنت استفاده می شود .
- توابع، دستوراتی که حاوی چند خط کد می باشند در بین{و} قرار میگیرند.
- دستوراتی که قبل از آنها # وجود دارد دستورات پیش پردازنده نام دارد که قبل از کامپایل شدن اجرا می شوند.

مفاهیم اولیه زبانC

- تعداد کلمات کلیدی در زبان C، 32 کلمه می باشد که در جدول زیر مشاهده می کنید.

کلمات کلیدی در C			
struct	int	double	auto
switch	long	else	break
typedef	register	enum	case
union	return	extern	char
void	signed	for	continue
while	static	if	do
volatile	sizeof	goto	default
unsigned	short	float	const

ساختار کد در زبان C

```
#include < name of the main library.h >

#include “name of the sub library.h”

#include “lib/name of the sub library.h”

Variables, constants, functions, declarations, macros and...  ;

Void main (void)

{

تعریف کدهایی که یک مرتبه اجرا میشوند;

مقدار دهی اولیه به رجیستر ها;

While(1)

{

دستوراتی که به صورت مداوم اجرا میشوند;

}

}
```

بخش دوم

آشنایی با انواع داده ها

داده

- داده:

به هر نوع اطلاعاتی گفته می شود که توسط برنامه مورد استفاده قرار میگیرد.

انواع داده:

1- داده صحیح integer ==> شماره دانشجویی

2- داده اعشاری floting point ==> نمره

3- داده کاراکتری character ==> نام و نام خانوادگی

4- داده بیت bit ==> پاس شدن

متغیرvariable

- متغیر:

محدوده ای مشخص از فضای حافظه SRAM با نام مشخص برای ذخیره داده، یک متغیر می تواند در محاسبات شرکت کند و یا نتیجه محاسبات را در خود حفظ کند.

انواع متغیر از نظر مکان:

متغیر های عمومی Global:

این متغیر ها قبل از تابع main تعریف می شوند و در کل برنامه می توان به آن دسترسی داشت.

متغیرهای محلیLocal:

این متغیر ها درون توابع تعریف میشوند و در خارج از آن تابع قابل دسترس نیستند.

جدول انواع متغیر

انواع متغیرها	اندازه بر حسب بیت	محدودهٔ تغییرات متغیر
bit, _Bit	1	1 , 0
bool, _Bool	8	1 , 0
char	8	to 127 128-
unsigned char	8	to 255 0
signed char	8	to 127 128-
int	16	to 32767 32768-
short int	16	to 32767 32768-
unsigned int	16	to 65535 0
signed int	16	to 32767 32768-
long int	32	to 2147483647 2147483648-
unsigned long int	32	to 4294967295 0
signed long int	32	to 2147483647 2147483648-
float	32	±1.175e-38 to ±3.402e38
double	32	±1.175e-38 to ±3.402e38

تعریف متغیر و مقدار دهی به آن

نوع متغیر نام متغیر;

int x;

نوع متغیر نام متغیر=مقدار;

int x=10;

نوع متغیر نام متغیر=مقدار, نام متغیر=مقدار;

int x=10, y=11, z=12;

نوع متغیر نام متغیر;

int x;

شیوه نامگذاری متغیر ها

- شروع نام متغیر نمی تواند با عدد باشد اما می تواند حاوی عدد باشد.
- نام متغیر نمی تواند از کلمات کلیدی رزرو شده باشد.
- طول نام متغیر نباید بیشتر از 31 کاراکتر باشد.
- نام متغیر نباید تکراری باشد.
- نام متغیر نمی تواند حاوی فاصله باشد، به جای استفاده از فاصله می توان از ___ ((خط زیرین)) استفاده کرد.
- نام متغیر نمی تواند حاوی کاراکتر هایی مثل #,$,=,+,&,^,!,%,@ و... باشد.
- بهتر است برای نامگذاری از حروف کوچک a تا z استفاده شود، از نام های مشخص و معلوم استفاده شود، از نکات ذهنی و دادن اطلاعات غلط خود داری کنید.

آرایهArray

- آرایه یک بعدی:
- به تعدادی از یک نوع متغیر با یک نام ثابت و اندیس های متفاوت که در کنار هم قرار میگیرند آرایه گفته میشود.

آرایه ها همانند یک لیست از اعداد عمل میکنند که می توان با قرار دادن یک متغیر به جای اندیس آن در سر تا سر این لیست حرکت کرد و در خانه های آن مقداری را قرار دهیم و یا از خانه های آن مقداری را برداریم.

تعریف آرایه و مقدار دهی به آن:

نوع آرایه نام آرایه [تعداد اعضا آرایه]; int x[3];

نام آرایه[شماره اندیس]=مقدار; x[0]=1, x[1]=2, x[2]=3;

نوع آرایه نام آرایه [تعداد اعضا آرایه]={مقدار اول, مقدار دوم, ... ,nمقدار اندیس}; int x[3]={1, 2, 3};

آرایهArray

- آرایه دو بعدی:
- یک ماتریس دارای سطر و ستون که هر خانه از ماتریس یک متغیر می باشد، اما با یک نام و اندیس های سطر و ستون متفاوت.

نوع آرایه نام آرایه [تعداد اعضا سطر ها] [تعداد اعضا ستون ها]={{مقادیر سطر اول},{مقادیر سطر دوم}};

```
int x[2][3]={{1, 2, 3},{4, 5, 6}};
```

رشته string

- رشته:

رشته همان آرایه معمولی اس که حاوی اطلاعات حروف می باشد، باید از نوع char باشد.

تعریف رشته و مقدار دهی به آن:

```
char   نام رشته[تعداد اعضا رشته]={’ حرف ’,’ حرف ‘,’ حرف ‘ ,…., ’ تهی ‘};

char  str[6]={‘H’,’e’,’l’,’l’,’o’};


char    نام رشته [ تعداد اعضا رشته]=”حروف“;

char  str[6]=“Hello”;
```

ثابت هاconstant

به یک نام مشخص که یک نام را به خود تعلق می دهد ثابت گفته می شود، ثابت ها برروی حافظه flash قرار میگیرند.

const نوع ثابت نام ثابت=مقدار;

const float pi=3.14;

با استفاده از نوع const میتوانیم آرایه ثابت تعریف کنیم.

#define نام ثابت مقدار

#define pi=3.14

ماکرو macro

به تغییر نام ذخیره شده در کامپایلر macro گفته می شود.

#define key PINB.7

#define نام جدید نام استاندارد

ماکرو macro

به تغییر نام ذخیره شده در کامپایلر macro گفته می شود.

#define نام جدید نام استاندارد

#define key PINB.7

بخش سوم

عملگرها و عملوندها

عملگر و عملوند

عملگر:

نمادی که عمل خاصی را انجام می دهد.

عملوند:

مقادیری که عملگرها برروی آنها عمل میکنند.

عملگرهای محاسباتی

عملگرهایی هستند که برای انجام عملیات های ریاضی مورد استفاده قرار میگیرند.

- 1-عملگر + جمع
- 2-عملگر – تفریق
- 3-عملگر * ضرب
- 4-عملگر / تقسیم
- 5-عملگر % باقیمانده تقسیم
- 6-عملگر – کاهش یک واحدی
- 7-عملگر ++ افزایش یک واحدی

عملگرهای رابطه ای

عملگرهایی هستند که برای مقایسه دو عبارت با یکدیگر مورد استفاده قرار میگیرند، نتیجه این عملگر ها یا صحیح و یا غلط می باشد. این عملگرها معمولا در شرط ها کاربرد دارند.

- 1-عملگر== اگر مقادیر دو عملوند باهم برابر باشند شرط درست است.
- 2-عملگر=! اگر مقادیر دو عملوند باهم برابر نباشند شرط درست است.
- 3-عملگر > اگر عملوند چپ بزرگتر از عملوند راست باشد شرط درست است.
- 4-عملگر < اگر عملوند چپ کوچکتر از عملوند راست باشد شرط درست است.
- 5-عملگر=> اگر عملوند چپ بزرگتر مساوی از عملوند راست باشد شرط درست است.
- 6-عملگر=< اگر عملوند چپ کوچکتر مساوی از عملوند راست باشد شرط درست است.

عملگرهای منطقی

عملگرهایی هستند که برای مقایسه شرایط وضع شده مورد استفاده قرار میگیرند، نتیجه این عملگر ها یا صحیح و یا غلط می باشد. این عملگرها معمولا در شرط ها کاربرد دارند.

- 1-عملگر&& عملگر AND ((و)) اگر هر دو قسمت برقرار باشند شرط درست است.
- 2-عملگر|| عملگر OR ((یا)) اگر حداقل یک قسمت از دو قسمت برقرار باشد شرط درست است.
- 3-عملگر! عملگر NOT ((نقیض)) اگر یک قسمت معکوس قسمت دیگری باشد شرط درست است.

عملگرهای بیتی

عملگرهایی هستند که برای اثر گذاری بر روی بیت ها مورد استفاده قرار میگیرند.

- 1- عملگر & عملگر AND ((و)) وقتی یک است که هر دو بیت یک باشد.
- 2- عملگر | عملگر OR ((یا)) وقتی صفر است که هر دو بیت صفر باشد.
- 3- عملگر ~ عملگر NOT ((نقیض)) بیت را معکوس میکند.
- 4- عملگر ^ عملگر XORبه ازای صفر یک می دهد و به ازای یک صفر.
- 5- عملگر >> عملگر شیفت به چپ مقدار عملوندچپ را به مقدار بیت های تعیین شده در عملوند راست به چپ شیفت می دهد.
- 6- عملگر << عملگر شیفت به راست مقدار عملوندراست را به مقدار بیت های تعیین شده درعملوند چپ به راست شیفت می دهد.

عملگرهای انتسابی

- 1- عملگر=مقدار عملوند سمت راست را در متغیر چپ قرار می دهد.
- 2- عملگر=+ مقدار عملوند سمت راست را با متغیر چپ جمع و در متغیر چپ قرار می دهد.
- 3- عملگر=- مقدار عملوند سمت راست را با متغیر چپ تفریق و در متغیر چپ قرار می دهد.
- 4- عملگر=* مقدار عملوند سمت راست را با متغیر چپ ضرب و در متغیر چپ قرار می دهد.
- 5- عملگر=/ مقدار عملوند سمت راست را با متغیر چپ تقسیم و در متغیر چپ قرار می دهد.
- 6- عملگر=>> انتساب شیفت به چپ.
- 7- عملگر=<< انتساب شیفت به راست.
- 8- عملگر=& انتساب AND.
- 9- عملگر=| انتساب OR.

عملگرهای کاربردی

- 1-عملگر() عملگر پرانتز اولیت ایجاد میکند.
- 2-عملگر , با استفاده از عملگر , میتوان در یک خط چند عمل را تعریف کرد.
- 3-عملگر؟ با استفاده از عملگر ؟ میتوان نسبت صحیح بودن و یا غلط بودن یک شرط از بین دو مقدار عملوند انتساب را انجام داد.

متغیر= شرط مورد نظر ? مقدار در زمان برقراری شرط : مقدار در زمان عدم برقراری شرط;

```
i= i>7    ?  0    :    1;
```

بخش چهارم

گذری بر سیستم اعداد

بیت و بایت

1Bit:0 , 1

1Byte=8Bit ==>

1024Byte=1kB,1024kB=1MB

سیستم اعداد

مبنا های پرکاربرد برای ما در زبان C مبنای دسیمال(10)، مبنای باینری(2) و مبنای هگزا دسیمال(16) می باشد.

پیشوند 0b نماد اعداد باینری و پشیوند 0x نماد اعداد هگزا دسیمال می باشد.

اگر مبنا را n در نظر بگیریم در هر مبنا تعداد اعداد از 0 تا n-1 می باشد.

مبنای هگزا دسیمال برای ما از سایر مبنا ها مهم تر می باشد چون:

1-تبدیل و بازی با بیت ها در این مبنا آسان تر است.

2- سریع تر به مقدار باینری دست می یابیم.

سیستم اعداد

اعداد هگزا دسیمال:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

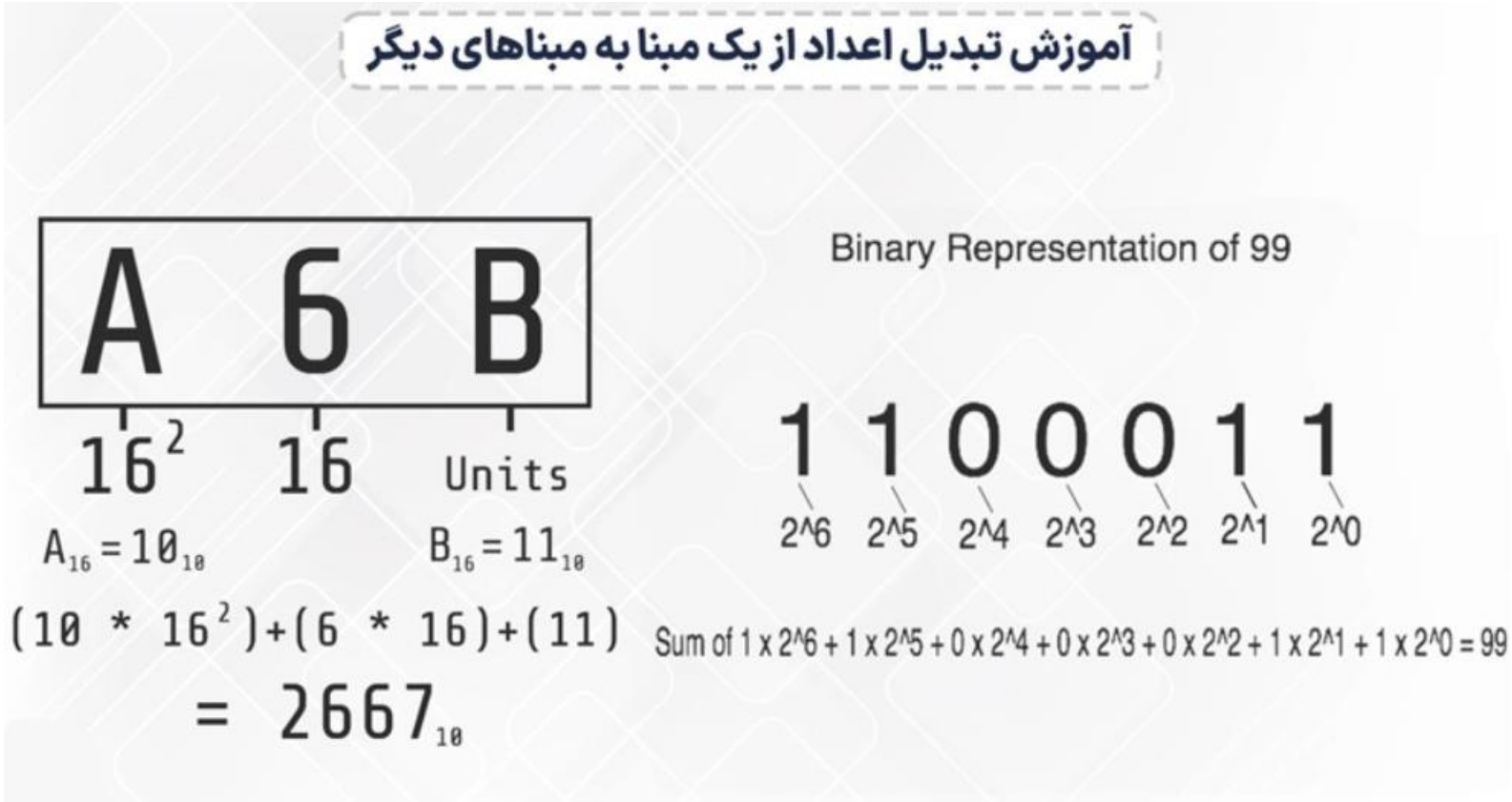
اعداد دسیمال:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9

اعداد باینری:

0, 1

سیستم اعداد


آموزش تبدیل اعداد از یک مبنا به مبناهای دیگر
A 6 B
16^2 16 Units
A16 = 1010
B16 = 1110
(10 * 16^2)+(6 * 16)+(11)
= 2667_10
Binary Representation of 99
1 1 0 0 0 1 1
2^6 2^5 2^4 2^3 2^2 2^1 2^0
Sum of 1 x 2^6 + 1 x 2^5 + 0 x 2^4 + 0 x 2^3 + 0 x 2^2 + 1 x 2^1 + 1 x 2^0 = 99

بخش پنجم

ساختار ها در زبان C

ساختار ها در زبان C

ساختار توالی:

از نقطه شروع دستورات را به ترتیب اجرا می کند و به پایان می رساند.

ساختار انتخاب:

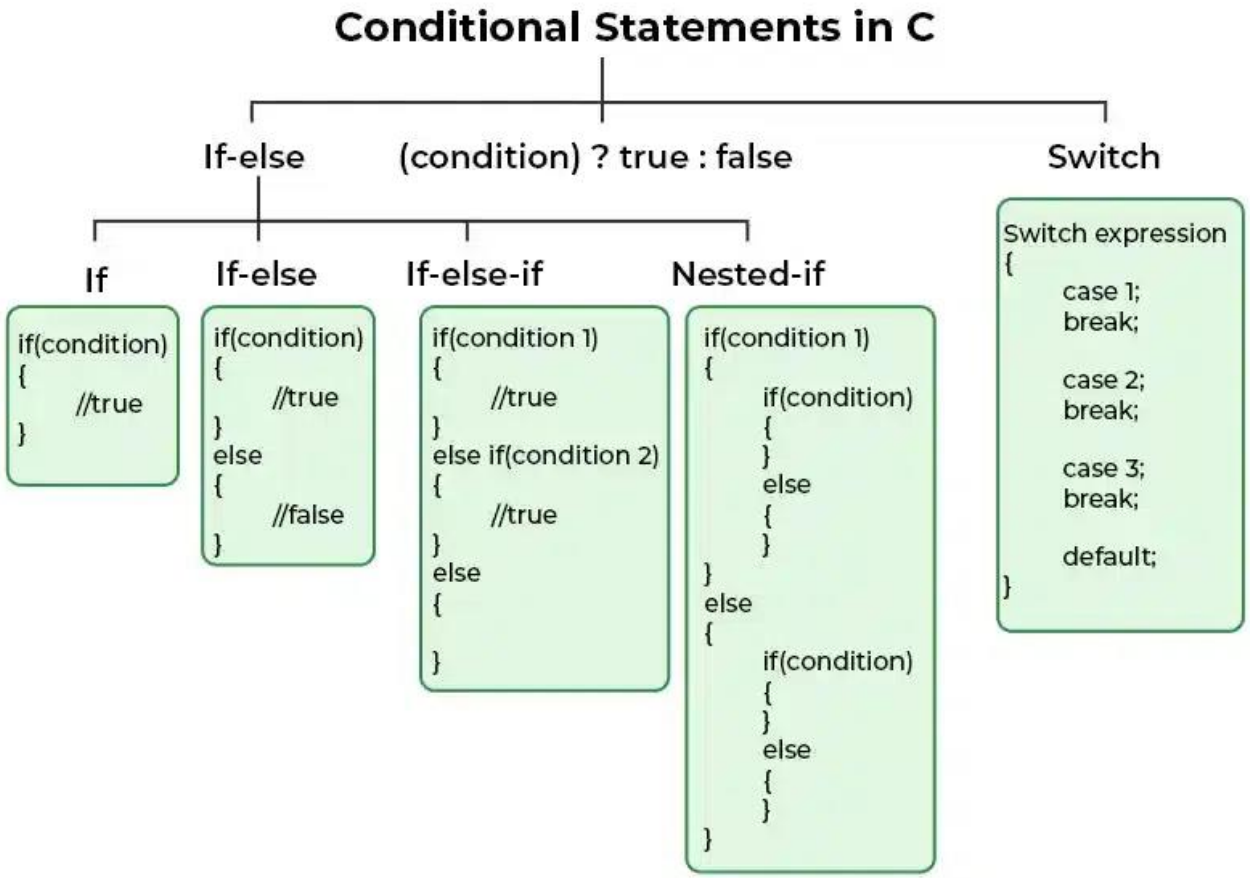
پیاده سازی دستورات شرطی و انتخاب جواب شرط.

ساختار تکرار:

بررسی و تکرار یک بلوک به تعداد مشخص.

دستورات شرطی

برنامه نویس میتواند cpu را موظف کند که یک شرط را بررسی کند و درصورت درست بودن جواب شرط، دستورات مربوط به آن شرط را انجام دهد.



دستورات شرطی

دستور شرطی if:

به وسیله دستور شرطیif می توانیم متغیرها آرایه ها رشته ها و... را مورد شرط قرار دهیم .

در دستور شرطی if عامل شرط بررسی می شود و اگر جواب شرط درست باشد دستورات مربوطه که کاربر تعیین کرده انجام می شود.

قالب دستور if:

```
if(                    شرط/شرط ها    ((اگر))    ){
دستور/دستورات((در زمان برقراری شرط اجرا می شود));
}
else   if(             شرط/شرط ها    ((درغیر این صورت))    ){
دستور/دستورات((در زمان برقرار نبودن شرط بالا و برقراری شرطelse if) );
}
else                   اگر شرط if و else if برقرار نباشد،    ((در نهایت))
{
دستور/دستورات;
}
```

```c
if (number>0){
        printf("the number is positive.");
}
else if(number<0){
        printf("the number is negative.");
}
else
{
        printf("the nimber is zero");
}
```

دستورات شرطی

دستور شرطی switch:

به وسیله دستور شرطیswitch می توانیم شروط برابری، عبارات صحیح و یا عبارات کاراکتری را با سرعت بالا مورد شرط قرار دهیم.در دستور شرطیswitch میتوانیم تصمیمات چند گانه بگیریم.

قالب دستور switch:

switch(عامل مورد شرط){

case مقدار اول :

کدهایی که درصورت برابر بودن عامل شرط با مقدار اول باید اجرا شوند;

break;

case n مقدار :

کدهایی که درصورت برابر بودن عامل شرط با مقدار n باید اجرا شوند;

break;

default:

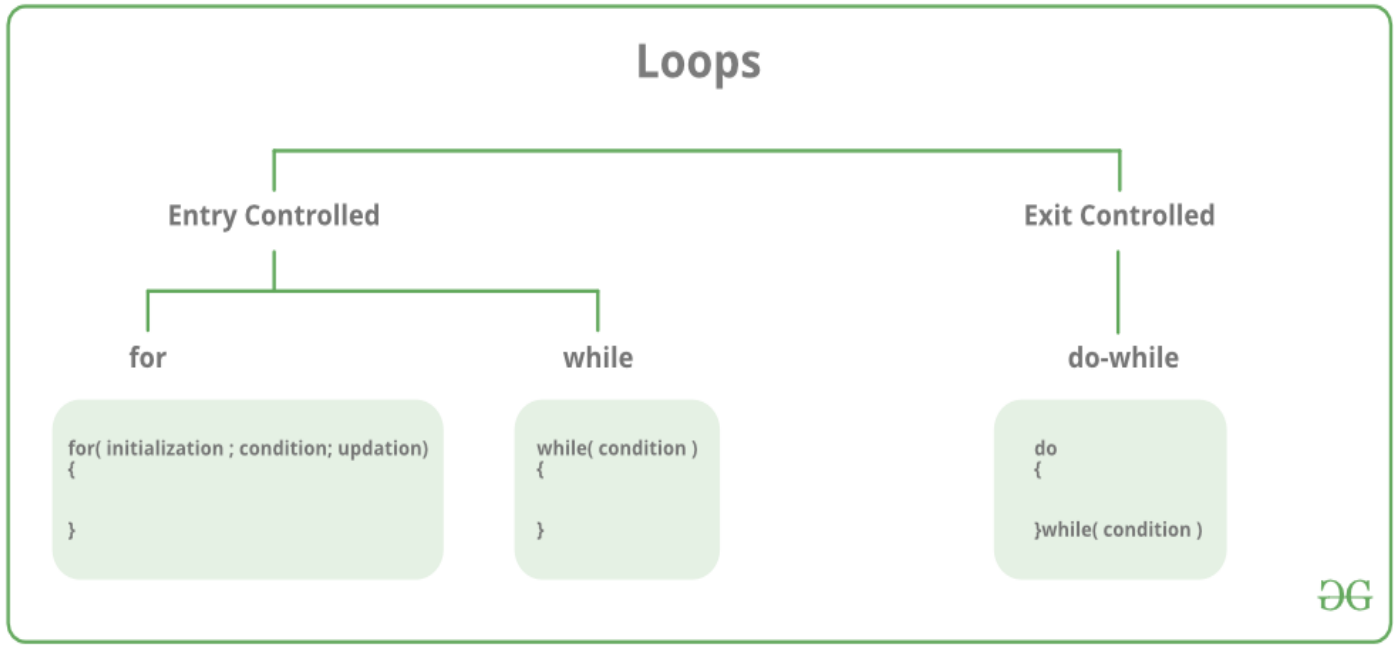
کدهایی که درصورت برابر نبودن عامل شرط با هیچ یک از مقادیر قید شده باید اجرا شوند;

}

```
switch(button){
        case 1 :
        printf("on");
        break;
        case 2:
        printf("off");
        break;
        case 3 :
        printf("setting");
        break;
        default :
        printf("no button pressed");
}
```

حلقه های تکرار

برنامه نویس میتواند cpu را موظف کند که تا زمانی که شرط حلقه برقرار باشد دستورات تکرار شونده مربوط به حلقه تکرار شوند و تا زمانی که حلقه در حال اجرا است برنامه در محل حلقه بماند.



حلقه های تکرار

حلقه while:

یکی از حلقه های تکرار در زبان C حلقه while است. حلقه while دارای یک شرط است که تا زمانی که این شرط برقرار باشد کد های مربوط به حلقه را تکرار میکند.

قالب حلقه while:

while(شرط حلقه){

کد هایی که تا زمان برقراری شرط حلقه تکرار می شوند;

}

```
while (a<16)
{
    PORTD=a;
    a++;
}
```

حلقه های تکرار

حلقه d0….while:

یکی از حلقه های تکرار در زبان C حلقه do …. while است. حلقه do …. while کاملا شبیه به حلقه while است اما در حلقه do …. while شرط حلقه در انتهای حلقه بررسی می شود، کد های حلقه حداقل یک مرتبه اجرا شده و در پایان شرط حلقه چک می شود و اگر شرط برقرار باشد تکرار کد های حلقه ادامه پیدا می کند.

قالب حلقه d0….while:

Do

{

کد هایی که تا زمان برقراری شرط حلقه تکرار می شوند;

}while(شرط حلقه);

```
do{
        PORTA=a;
        a++;
}while (a<16)
```

حلقه های تکرار

حلقه for:

حلقه for را می‌توان مهمترین حلقه در زبان C دانست، در حلقه for تعداد تکرار و شرط پایان مشخص می‌شود و دارای یک متغیر شمارشی است. از حلقه for برای اجرای یک بلوک کد بر اساس شرایط و محدودیت‌های مشخص شده و تعداد دفعات تکرار مشخص، استفاده می‌شود.

قالب حلقه for:

for(گام حرکت ; شرط تکرار حلقه ; مقدار اولیه=متغیر کنترلی)

{

کد هایی که تا زمان برقراری شرط حلقه تکرار می شوند;

}

```
for ( i = 0; i < 8; i+=5)
{
    a[i]=i;
}
```

حلقه های تکرار

دستور goto:

با استفاده از دستور goto می توان از نقطه ای به نقطه دیگری پرش کرد یا به عبارتی اجرای برنامه را می توان به هر قسمت دلخواهی انتقال داد.

قالب دستور goto:

برچسب:

...

goto برچسب;

```
x1:
....
goto x1;
```

حلقه های تکرار

دستور break:

برنامه با دیدن دستور break در هر نقطه از حلقه که باشد از آن نقطه در حلقه خارج شده و کد های زیر حلقه را اجرا می کند.

قالب دستور break:

....

break;

حلقه های تکرار

دستور continue:

برنامه با دیدن دستور continue در هر نقطه از حلقه که باشد کد های زیر دستورcontinue را رها کرده و به ابتدای حلقه باز میگردد.

قالب دستور continue:

....

continue;

حلقه های تکرار

نکات مربوط به حلقه ها:

1- while(1)حلقه بینهایت است.

2- while(a==16); یعنی برنامه تا زمانی که a برابر با 16 نشده است در این خط گیر کند.

3-for(;;)حلقه بینهایت است.

بخش پنجم

توابع در زبان C

توابع

تابع مانند دستگاهی است که مواد اولیه را دریافت می کند عمل مورد نظر را بر روی مواد اولیه انجام می دهد و در خروجی محصول مطلوب را تحویل می دهد.

انواع توابع:

1-توابع پیش تعریف شده یا توابع کتابخانه ای.

2-توابعی که توسط برنامه نویس تعریف می شود.

هر تابع براساس هدفی که نوشته می شود دارای ورودی و خروجی است ورودی تابع را آرگومان تابع و خرجی تابع را مقدار برگشتی تابع می نامیم.

توابع

انواع توابع از نظر ورودی و خروجی:

1-تابع با ورودی با خروجی

2-تابع با ورودی بدون خروجی

3-تابع بدون ورودی با خروجی

4-تابع بدون ورودی بدون خروجی

توابع

ناحیه تعریف توابع:

1-در بالای تابعmain

2-در پایین تابعmain حتما باید در بالای تابع main اعلان شود==> (معرفی ورودی تابع) نام تابع نوع داده خروجی

3-در فایل های کتابخانه

توابع

تابع با ورودی با خروجی

نوع تابع نام تابع (معرفی ورودی ها و نوع ورودی های تابع)

{

معرفی متغیر های محلی;

کد های تابع;

return مقداری که باید به خروجی تابع باز گردد;

}

```
int multiply (int a, int b)
{
    int result;
    result= a*b;
    return result;
}



    x=multiply(4,5);
    y=multiply(i,j);
```

توابع

تابع با ورودی با بدون خروجی

void نام تابع (معرفی ورودی ها و نوع ورودی های تابع)

{

معرفی متغیر های محلی;

کد های تابع;

}

```
void and (int x,int y)
{
      int result ;
      result= x & y ;
      PORTD = result;
}



and(4,2);
and(i,j);
```

می توان مقادیر دریافت شده از ورودی را بر روی رجیسترهای مورد نظر قرار داد.

توابع

تابع بدون ورودی با خروجی

نام تابع (void) نوع داده خروجی

{

معرفی متغیر های محلی;

کد های تابع;

return مقداری که باید به خروجی تابع باز گردد;

}

```
int a_and_b (void)
{
    int a,b,c;
    a=PINA.0;
    b=PINA.1;
    c=a|b;
    return c;
}

PORTC= a_and_b();
```

می توان مقادیر قرارگرفته برروی رجیسترها را خواند و به خروجی باز گرداند.

توابع

- تابع بدون ورودی بدون خروجی

```
void led_mod (void)
{
        PORTB=0xf0
        PORTC=0x0F;
}

led_mod();
```

```
void  نام تابع  (void)
{
معرفی متغیر های محلی;
کد های تابع;
}
```

می تواند عملیاتی را بر روی رجیستر ها انجام دهد.

بخش پنجم

ساختمان،یونیون و نوع شمارسی

ساختمان،یونیون و نوع شمارسی

کلاس های حافظه:

ویژگی از متغیرها که شان می‌دهد یک متغیر در چه قسمت هایی از برنامه قابل دسترس است و مدت زمانی که متغیر در حافظه وجود دارد را چقدر است.

1-کلاس حافظه اتوماتیک Automatic: متغیرهای محلی که داخل تابع تعریف می‌شوند با تابع شروع و با تابع به پایان می‌رسند و مقداری را به خود اختصاص می‌دهند.

2-کلاس حافظه ثبات register: متغیرهای محلی را در ثبات‌های پردازنده قرار می‌دهد و باعث سرعت بیشتر می‌شود.

3-کلاس حافظه استاتیک static: هنگام فراخوانی تابع ایجاد می‌شوند و هنگام خروج از تابع آخرین مقدار خود را حفظ می‌کنند.

4-کلاس حافظه خارجی extern: متغیرهایی که خارج از توابع تعریف می‌شوند و به کامپایلر اعلام می‌کنند که یک مرتبه برای آن متغیر حافظه اشغال کند حتی در فایل‌های مختلف.

ساختمان،یونیون و نوع شمارسی

قالب مشخص کردن کلاس حافظه یک متغیر:

کلاس حافظه نوع متغیر نام متغیر;

```
        int         x;
reggister int       y;
static    int       z;
extern    int       w;
```

ساختمان،یونیون و نوع شمارسی

اشاره گرpointer:

اشاره گر مانند یک متغیر است گه به جای نگهدار مقدار آدرس حافظه را نگهداری می کند.

هر اشاره گر می تواند آدرس متغیر هم نو خود را در خود نگه دارد.

عملگر های اشاره ای:

: *

تعریف اشاره گر

دسترسی به محتوای اشاره گر

:&

دسترسی به آدرس

ساختمان،یونیون و نوع شمارسی

قالب اشاره گرpointer:

```
نوع اشاره گر      *نام اشاره گر;
    int              *y;
    int              *x,y,z;
    ….
    y=142;
    x=&y;
    z=*x;
```

ساختمان،یونیون و نوع شمارسی

ساختمان struct:

ساختمان می‌تواند به ما این امکان را بدهد که داده‌ها و متغیرها را در یک واحد منطقی و جامع گروه‌بندی کنیم می‌توانیم ساختاری از انواع داده را در قالب یک نام داشته باشیم برای استفاده از عناصر ساختمان معرفی شده باید متغیرها از نوع ساختمان پس از آن معرفی شود.

ساختمان،یونیون و نوع شمارسی

قالب ساختمان struct:

struct نام ساختمان;

{

عناصر ساختمان;

}نام متغیرها;

....

نام عنصر مورد نظر در ساختمان . نام متغیر از نوع ساختمان;

```
struct person
{
        char name [20];
        int age;
        float height;
}person1,person2;
```

```
person1.age=25;
```

توابع

union:

union محلی از حافظه است که توسط دو یا چند متغیر، به صورت اشتراکی استفاده می‌شود.

متغیرها همزمان نمی‌توانند از این محل حافظه استفاده کنند.

union طول بیشترین متغیر در خود به لحاظ بی تی را به عنوان طول خود در نظر می‌گیرد.

ساختمان،یونیون و نوع شمارسی

قالب union:

union نام یونیون

{

union عناصر;

} ;

....

union unionنام نام متغیرها;

نام متغیر. union در نظر مورد عنصر نام=مقدار;

```
union MYUnion
{
        int myInt;
        float myFloat;
        char mychar;
};

....
union MYUnion u;

u.myInt=42;
```

ساختمان،یونیون و نوع شمارسی

:enum

یک نوع داده در زبان C است که برای تعریف مجموعه‌ای از مقادیر ثابت استفاده می‌شود.

در enum هر عنصر با یک نام و یک مقدار ثابت تعریف می‌شود، نام‌ها معمولاً با حروف بزرگ نوشته می‌شوند و مقادیر ثابت به صورت پیش فرض از 0 شروع می‌شوند.

ساختمان،یونیون و نوع شمارسی

قالب enum:

enum enum نام

{

عنصر اول

عنصر دوم

عنصرسوم

nعنصر

}متغیر ;

....

enum enum نام نام متغیرها=مقدار;

```
enum Color
{
        RED,
        GREEN,
        BLUE
};

 enum Color c1 = RED;
enum Color c2 = GREEN;
```