نوشتن COMMENT

اگر توضیحات ما بیش از یک خط باشد از */ و /* استفاده می کنیم.

```
/*this program
  is written by .....
  Date: 1384/9/13
*/
```

کامپایلر خطوطی را که بین */ و /* قرار گرفته اند را در نظر نمی گیرد.

دستورات تکرار (حلقه ها)

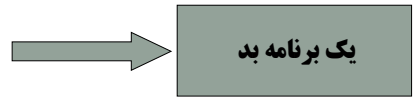
- **با استفاده از دستورات تکرار (حلقه ها) می توانیم اجرای بخشی از برنامه را چند بار تکرار کنیم.**

- **سه دستور تکرار در زبان C**
  - for
  - while
  - do-while

ضرورت وجود حلقه ها در برنامه

- **مثال ۱) برنامه ای بنویسید که اعداد ۱ تا ۱۰۰ را چاپ کند.**

```
#include <stdio.h>
void main()
{
  int i;
   i=0;
   i++;
   printf("%d",i);
   i++;
   printf("%d",i);
   .....
   .....
   ...
}
```

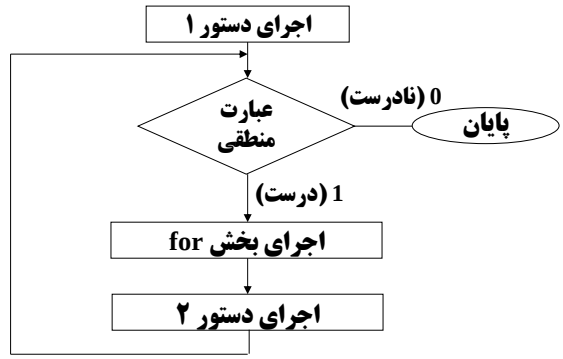


دستور FOR

for (دستور ۱ ;عبارت منطقی ;دستور ۲)

دستور(ات)

بخش for: بخشی که اجرای آن تکرار خواهد شد.
برای بیش از یک دستور در بخش for استفاده از { } لازم است.



مثال

- **مثال ۲) عملکرد قطعه برنامه زیر را توضیح دهید:**

```
for (i=0;i<2;i++)
    printf("\n%d" ,i);
```

1. i=0
2. چون i<2 است دستور printf اجراء می شود و عدد 0 در مانیتور چاپ می شود.
3. i++ اجرا می شود. در نتیجه i=1
4. i=1<2 در نتیجه مجددا دستور printf اجراء می شود و این بار عدد 1 در مانیتور چاپ می شود.
5. i++ اجرا می شود. در نتیجه i=2
6. این بار i=2<2 برقرار نیست. در نتیجه از حلقه خارج می شویم.

دستور printf دو بار اجرا شد.

- مثال ۳) مثال ۱ با استفاده از حلقه ها

```
#include <stdio.h>
void main()
{
  int i=0;
  for (i=1;i<=100;++i)
        printf("\n%d",i);
}
```

چند مثال دیگر

for (i=5;i<=8;i++) printf(“\n%d” ,i);	for (x=5;x>8;x++) printf(“\n%d” ,x); حلقه اصلا اجرا نمی شود
for (k=8;k>5;k--) printf(“\n%d” ,k);	for (num=3;num>0;num++) printf(“\n%d” ,num); حلقه پایان ناپذیر (بی نهایت)
for (i=8;i>1;i/=2) printf(“\n%d” ,i);	

مثال

- عملکرد قطعه برنامه زیر:

```
scanf(“%d”,&i);
if (i>0)
   for(j=0;j<i;++j)
        printf(“\n%d”,j);
```

مشابه if ، for و دستورات داخل حلقه مجموعا یک دستور فرض می شوند و در نتیجه نیازی به { } برای if نیست.

مثال

- برنامه ای بنویسید که ۱۰۰ عدد را از کاربر بگیرد و حاصل جمع آنها را چاپ کند.

```
#include <stdio.h>
void main()
{
  int i;
  float sum;
  float x;
  sum=0;
  for (i=0;i<100; i++)
  {
   scanf("%f", &x);
   sum+=x;
  }
  printf("The result=%f",sum);
}
```

حلقه های تودرتو(NESTED FOR)

- **عملکرد برنامه زیر را توضیح دهید.**

```
for (i=0;i<3;++i)
  for (j=0;j<2;++j)
       printf(“*”);
```

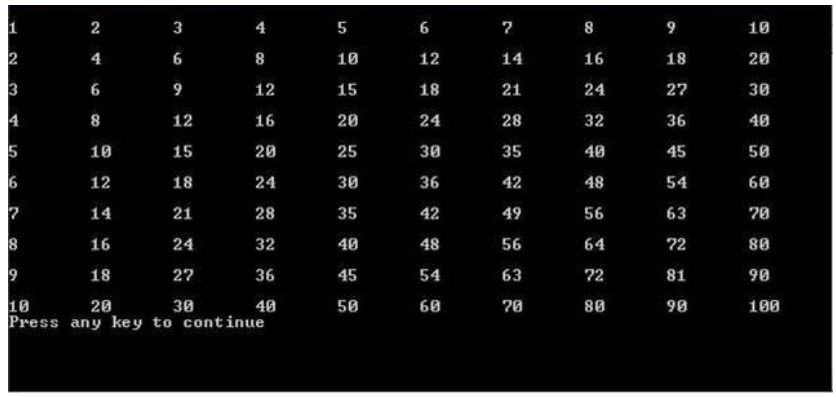
حلقه ای که در درون آن حلقه ای دیگر قرار گرفته است.
for اول ۳ بار اجرا می شود و در هر بار اجرای آن for دوم، دو بار اجرا می شود. در نتیجه ۶ علامت ستاره در خروجی چاپ خواهد شد.

مثال

- **عملکرد برنامه زیر:**

```
#include <stdio.h>
void main()
{
    int i,j;
    for (i=1;i<=10;++i)
    {
       printf("\n");
       for (j=1;j<=10;++j)
            printf("%d\t",(i*j));
    }
}
```

چاپ جدول ضرب ۱۰*۱۰



نکاتی در مورد دستور FOR

نکته ۱: در بخش های دستور ۱ و دستور ۲ هر دستوری می تواند باشد.

مثال

```
#include <stdio.h>
#include <conio.h>
void main()
{
   char ch;
   float x,sum=0;

   for (ch='y';ch=='y';ch=getche())
   {
         printf("\nEnter a number:");
         scanf("%f",&x);
         sum+=x;
         printf("To continue press y: ");
   }

   printf("\nThe sum=%f",sum);
}
```

برنامه تا زمانی که کاربر y را فشار می دهد عدد می خواند و زمانی که کاربر کلیدی غیر از y را فشار دهد مجموع اعداد گرفته شده را چاپ می کند.

نکاتی در مورد دستور FOR(ادامه)

نکته ۲) هر یک از بخش های دستور ۱ ، دستور ۲ یا عبارت منطقی را می توان حذف کرد

مثال ۱

```
#include <stdio.h>
void main()
{
   int i=3;
   for (;i<7;++i)
  {
      printf("%d\t",i);
  }
}
```

مثال ۲

```
#include <stdio.h>
void main()
{
   int i;
   for (i=3;i<7; )
  {
      printf("%d\t",i);
      i++;
  }
}
```

مثال ۳

```
#include <stdio.h>
void main()
{
   int i;
   for (i=3;;i++ )
  {
      printf("%d\t",i);
  }
}
```

نکته ۳) در بخش های دستور ۱ و ۲ می توان بیش از یک دستور قرار داد. دستورات با استفاده از کاما (,) از هم جدا می شوند.

مثال:

```
#include <stdio.h>
void main()
{
  int m,n;
  for (m=1,n=8; m<n;m++,n--)
        printf("m=%d , n=%d\n",m,n);
}
```

دستور WHILE

while (عبارت مقایسه ای)
دستور(ات)

- **عملکرد: اگر عبارت مقایسه ای درست باشد دستور(ات) اجرا می شود. اگر عبارت مقایسه ای نادرست باشد از حلقه خارج می شویم.**
- **استفاده از {} در صورت وجود بیش از یک دستور در حلقه**

مثال

```
i=0;
while (i<4)
{
    printf("%d",i);
    i++;
}
```

عبارت 0123 روی مانیتور نشان داده می شود

مثال

```
#include <stdio.h>
void main()
{
  char ch;
  int num,sum;
  ch='y';
  sum=0;
  while (ch=='y')
  {
    printf("\nEnter a number: ");
    scanf("%d",&num);
    sum+=num;
    printf("\ncontinue(enter y/n):");
    scanf("\n%c",&ch);
  }
  printf("The result= %d",sum);
}
```

تا زمانی که کاربر y **را انتخاب کند اعداد گرفته می شود**
با انتخاب هر کلیدی غیر از y **از حلقه خارج می شویم**
و حاصل جمع مقادیر چاپ می شود.

نکته–تفاوت مهم FOR و WHILE

- **با استفاده از دستور** while **می توان حلقه هایی نوشت که در آنها تعداد تکرار از قبل مشخص نیست. برخلاف** for **که معمولا در آن تعداد تکرار از قبل مشخص است.**

دستور DO-WHILE

do
دستور(ات)
while (**عبارت مقایسه ای**);

عملکردی کاملا مشابه while **دارد. با این تفاوت که در** while **عبارت مقایسه ای در ابتدا و قبل از اجرای دستورات چک می شود، اما در** do-while **این کار در انتها و بعد از اجرای دستورات انجام می شود.**
بخش حلقه لااقل یک بار اجرا خواهد شد.

مثال

```
#include <stdio.h>
void main()
{
  char ch;
  int num,sum;
  sum=0;
  do
  {
    printf("\nEnter a number: ");
    scanf("%d",&num);
    sum+=num;
    printf("\ncontinue(enter y/n):");
    scanf("\n%c",&ch);
  }
  while (ch=='y');
 printf("The result= %d",sum);
}
```

دستور BREAK

- موجب خروج از حلقه می شود.
- مثال :

```
#include <stdio.h>
void main()
{
    int t;
    for (t=0;t<100;t++)
    {
        printf("%d",t);
        if (t==5)
            break;
    }
}
```

چند مثال برنامه نویسی

- برنامه ای بنویسید که ضرایب یک معادله درجه ۲ را بگیرد و ریشه های آن را محاسبه کند.

$$ax^2 + bx + c = 0$$

$$\Delta = b^2 - 4ac$$

$$\begin{cases} x_{1,2} = \dfrac{-b \pm \sqrt{\Delta}}{2a} & \Delta > 0 \\ x_1 = \dfrac{-b}{2a} & \Delta = 0 \\ no \quad root & \Delta < 0 \end{cases}$$

1. حل مسئله:
2. الگوریتم:
   1. گرفتن ضرایب a، b و c
   2. محاسبه دلتا
   3. محاسبه ریشه ها بر اساس مقدار دلتا

نکات:

• دستور sqrt برای محاسبه ریشه دوم به کار می رود.

header file آن ، math.h می باشد.

```
#include <stdio.h>
#include <math.h>
void main()
{
   float a,b,c,x1,x2,delta;
   printf("Enter the coefficients");
   printf("\na=");
   scanf("%f",&a);
   printf("b=");
   scanf("%f",&b);
   printf("c=");
   scanf("%f",&c);
   delta=b*b-4*a*c;
   if (delta>0)
   {
         x1=(-b+sqrt(delta))/(2*a);
         x2=(-b-sqrt(delta))/(2*a);
         printf("x1=%.2f\tx2=%.2f",x1,x2);
   }
   else
         if (delta==0)
         {
                  x1=-b/(2*a);
                  printf("x1=%.2f",x1);
         }
         else
                  printf("No root");
}
```

ادامه چند مثال برنامه نویسی

- **بازی حدس: توسط کامپیوتر عدد صحیح تصادفی بین ۰ تا ۱۰۰۰تولید می شود (لکن به کاربر نشان داده نمی شود) سپس از کاربر خواسته می شود عددی را به عنوان حدس وارد کند . اگر عدد وارد شده مساوی عدد مطلوب باشد بازی با برنده شدن فرد خاتمه می یابد و اگر کوچکتر یا بزرگتر باشد به کاربر اعلام می شود. کاربر می تواند تا حداکثر ۱۳ حدس داشته باشد. در صورتی که تمام حدس ها اشتباه باشد کاربر بازنده می شود**

ادامه چند مثال برنامه نویسی

- **الگوریتم:**
  1. **تولید یک عدد بین ۰ تا ۱۰۰۰**
  2. **دریافت حدس کاربر**
  3. **اگر عدد کاربر برابر عدد مطلوب بود اعلام موفقیت**
  4. **و گرنه اعلام کوچکتر یا بزرگتر بودن به کاربر و سپس بازگشت به ۲**
  5. **انجام ۲ تا ۴ به تعداد ۱۳ بار**

rand() اعداد تصادفی بین ۰ تا ۳۲۷۶۷ تولید می کند. در نتیجه rand()%1001 اعداد تصادفی بین ۰ تا ۱۰۰۰ تولید می کند.

```
#include <stdio.h>
#include <stdlib.h>
void main()
{ int x,y,i;
  x=rand()%1001;

  for(i=0;i<13;i++)
   {
     printf("\n Guess %dth number:",(i+1));
     scanf("%d",&y);   if (x==y)
                       {
                       printf("\nyou win");
                       break;
                       }
      if(x>y)
        printf("\nthe number is greater than you entered");
      if(x<y)
        printf("\nthe number is less than you entered");
   }

 if(i==13)
    printf("you loss");
}
```

```
#include <stdio.h>
#include <math.h>
void main()
{
        float x,y,r,teta;
        printf("Enter (x,y)\n");
        printf("x=");
        scanf("%f",&x);
        printf("y=");
        scanf("%f",&y);
        r=sqrt(x*x+y*y);
        if ((x>0 && y>=0) || (x>0 && y<0))
                teta=atan(y/x);
        if (x<0 && y>0)
                teta=atan(y/x)+3.141592;
        if (x<0 && y<=0)
                teta=atan(y/x)-3.141592;
        if (x==0 && y>0)
                teta=3.141592/2;
        if (x==0 && y<0)
                teta=-3.141592/2;
        printf("polar cordinate:\n");
        printf("r=%.2f",r);
        printf("\nteta=%.2f radian\n",teta);
}
```

نکته: atan ، برای محاسبه arc tan به کار می رود. header file آن math.h است.

- **مثال) تبدیل مختصات دکارتی (x,y) به مختصات قطبی** (r,θ) $\theta \in [-\pi,\pi)$
  - **محاسبه r به سهولت قابل انجام است:** $r=\sqrt{x^2+y^2}$
  - **برای محاسبه وضعیت های مختلف زیر را خواهیم داشت:**
    - **(x>0 و y>=0) یا (x>0 و y<0)** $\theta=\arctan(\frac{y}{x})$
    - **(x<0 و y>0)** $\theta=\pi+\arctan(\frac{y}{x})$
    - **(x<0 و y<=0)** $\theta=-\pi+\arctan(\frac{y}{x})$
    - **(x=0 و y>0):** $\theta=\frac{\pi}{2}$
    - **(x=0 و y<0):** $\theta=-\frac{\pi}{2}$

الگوریتم:
1. **گرفتن (x,y)**
2. **محاسبه r**
3. **محاسبه θ با توجه به محل نقطه**