مبانی برنامه‌نویسی

مدرس: سعدون عزیزی

s.azizi@uok.ac.ir

مرکز آموزش‌های الکترونیکی

تابستان ۹۶

سرفصل مطالب

- آشنایی با کامپیوتر و الگوریتم
- مقدمه‌ای بر برنامه‌نویسی C
- محاسبات
- ورودی/خروجی
- حلقه ها
- دستورات شرطی
- توابع
- آرایه‌ها
- کاراکترها و رشته‌ها
- اشاره‌گرها
- ساختار
- فایل‌ها

انواع دستورها

- به طور کلی دستورهای زبان C را می توان به سه دسته تقسیم نمود:

- **دستور ساده:** دستوری که از یک عبارت تشکیل شده و به علامت ; ختم می شود

```
x=a+b;        یا        printf(“%d”,x);
```

- **دستور مرکب (بلوک):** به مجموعه دستورهایی گفته می شود که میان دو علامت { و } قرار می گیرند

```
{
x=a+b;
printf(“%d”,x);
}
```

- **دستور کنترلی:** دستوری است که در مورد دستور اجرایی بعدی تصمیم گیری می کند. در زبان C، ۶ نوع دستور کنترلی وجود دارد؛ ۳ دستور کنترل تکرار **for**، **while** و **do … while**، و ۳ دستور شرطی **if**، **if … else** و **switch**

عملگرهای تصمیم گیری

- **عملگرهای مقایسه ای:** در C، چهار عملگر برای مقایسه دو عبارت نسبت به یکدیگر وجود دارد: عملگر کوچکتری (<)، کوچکتر یا مساوی (=<)، بزرگتری (>) و بزرگتر یا مساوی (=>)

- اگر جواب معادل بله (true) باشد، حاصل عبارت 1 می شود، و در صورتی که جواب معادل خیر (false) باشد، حاصل عبارت 0 خواهد بود.

مثال: فرض کنید x=6 و y=2 باشد. در این صورت:

x>=y → 1

(x-2)<(y*3-2) → 0

عملگرهای تصمیم گیری

- **عملگرهای تساوی:** در C برای تساوی از عملگر == و برای عدم تساوی (مخالف بودن) از عملگر =! استفاده می شود.

- اگر جواب معادل بله (true) باشد، حاصل عبارت 1 می شود، و در صورتی که حاصل عبارت خیر (false) باشد، حاصل عبارت 0 خواهد بود.

مثال: فرض کنید x=6 و y=2 باشد. در این صورت:

x==y → 0

x-2 != 6-2*y → 1

عملگرهای تصمیم گیری

- **عملگرهای منطقی:** در C، سه عملگر منطقی && (and)، || (or) و ! (not) وجود دارند

- عملگر && یک عملگر دو طرفه است؛ چنانچه مقدار هر دو طرف true باشد حاصل برابر 1 و در غیر این صورت برابر 0 است.
- در عملگر دو طرفه || کافی است که مقدار یکی از طرفین true باشد تا حاصل 1 شود. در غیر این صورت، جواب عبارت 0 می شود
- عملگر ! یک عملگر یک طرفه است که نتیجه ارزیابی عبارت را عوض می کند؛ اگر مقدار عبارت true باشد حاصل برابر 0 و در غیر این صورت حاصل برابر 1 خواهد بود.

مثال: فرض کنید x=6 و y=2 باشد. در این صورت:

x<10 && y>0 → 1 x>5 || y>5 → 1 !(x>6 || y>=2) → 0

دستور while

- برای آن که یک فرآیند در برنامه، تا زمانی که شرطی برقرار است، تکرار شود از دستور while استفاده می گردد.

while (عبارت تصمیم گیری)
دستور

مثال: تکه برنامه زیر، ابتدا اعداد صحیح ۱ تا ۱۰ را تولید و چاپ می کند. سپس مقدار نهایی متغیر i چاپ می گردد.

```
i=1;
while (i<=10)
{
        printf("%d ",i);
        i++;
}
printf("\nThe value of i at this point is: %d",i);
```

خروجی:

```
1 2 3 4 5 6 7 8 9 10
The value of i at this point is: 11
```

دستور do … while

- دستور دیگری که برای تکرار یک فرآیند در برنامه، می توان از آن استفاده کرد، دستور do … while است.

نکته: دستور داخل حلقه do … while حداقل یک بار اجرا می گردد.

do

دستور

while (عبارت تصمیم گیری);

مثال: تکه برنامه زیر، ابتدا اعداد صحیح ۱ تا ۱۰ را تولید و چاپ می کند. سپس مقدار نهایی متغیر i چاپ می گردد.

```
i=1;
do
{
        printf("%d ",i);
        i++;
} while (i<=10);
printf("\nThe value of i at this point is: %d",i);
```

خروجی:

```
1 2 3 4 5 6 7 8 9 10
The value of i at this point is: 11
```

دستور for

- دستور for یکی دیگر از دستورهای پرکاربرد جهت ایجاد حلقه تکرار است.

for (مقداردهی اولیه;عبارت تصمیم گیری;عبارت به روز کننده)
دستور

مثال: تکه برنامه زیر، تمام اعداد فرد مثبت و کوچکتر از ۱۰۰ را تولید و چاپ می کند.

```
int i;
for (i=1; i<100; i+=2)
        printf("%d ", i);
```

خروجی:

1 3 5 7 9 11 … 99

ایجاد حلقه بی نهایت

- به کمک دستوارت زیر می توان یک حلقه بی نهایت ایجاد کرد:

```
for ( ; ; )
      دستور
```

یا

```
while (1)
      دستور
```

نکته: برای خروج از یک حلقه می توان از دستور break استفاده کرد که در فصل بعد آن را توضیح خواهیم داد.

مثال: دستور زیر ایجاد یک حلقه بی نهایت می کند:

```
int i;
for ( ; ; )
      printf("%d ", i++);
```

عملگر کاما (,)

- به طور کلی، می توان چند عبارت را با استفاده از عملگر کاما به دنبال هم نوشت و به عنوان یک عبارت در نظر گرفت.

مثال: به تکه برنامه های زیر دقت کنید

```
for (sum=0.0, i=1; i<=10; i++)
{
      scanf("%f", &x);
      sum+=x*x;
}
```

خروجی: این برنامه ۱۰ عدد اعشاری از ورودی دریافت می کند و جمع مجذور آنها را محاسبه می کند.

```
for (i=1, j=10; i<=5 && j>i; i++ , j--)
      printf("%d   %d\n", i,j);
```

خروجی:

```
1    10
2    9
3    8
4    7
5    6
```

حلقه های تودرتو

- در بدنه دستورهای کنترل تکرار (حلقه ها)، هر دستور دیگری می تواند قرار بگیرد از جمله یک دستور کنترل تکرار دیگر.

مثال: تکه برنامه زیر، با استفاده از حلقه های تودرتو، قسمتی از جدول ضرب را می سازد:

```
for (i=1; i<=3; i++)
{
    for (j=1; j<=4; j++)
        printf("%5d", i*j);
    printf("\n");
}
```

خروجی:

```
  1   2   3   4
  2   4   6   8
  3   6   9  12
```