مبانی کامپیوتر و برنامه‌سازی

یک برنامه ساده C

- چاپ یک خط متنی در خروجی

```
1  /* Fig. 2.1: fig02_01.c
2     A first program in C */
3  #include <stdio.h>
4
5  /* function main begins program execution */
6  int main( void )
7  {
8     printf( "Welcome to C!\n" );
9
10    return 0; /* indicate that program ended successfully */
11 } /* end function main */
```

توضیح نویسی

- روش توضیح نویسی // و /* */
  - // برای یک خط توضیح استفاده می شود.
  - /* */ برای چند خط توضیحات استفاده می شود.
- توضیحات به زبان ماشین ترجمه نمی‌شوند.
- توضیحات صرفا به خوانایی برنامه کمک می کند و عدم وجود توضیحات در اجرای برنامه اخلالی ایجاد نمی کند.

سرآیند برنامه

#include<stdio.h>

- راهنمایی برای پیش‌پردازشگر است.
- خطوطی که علامت # دارند، قبل از اینکه برنامه کامپایل شود توسط پیش‌پردازشگر استفاده می‌شوند.
- این خط به پیش‌پردازشگر می‌گوید که محتوی فایل سرآیند ورودی/خروجی (<stdio.h>) را در برنامه بگنجاند.
- این فایل سرآیند شامل اطلاعاتی است که توسط کامپایلر استفاده می‌شود.
  - اگر اجرا فراخوانی‌ای به توابع ورودی یا خروجی مانند printf داشته باشد نیاز است از این هدر استفاده شود.

بدنه اصلی برنامه

int main(void)

- این عبارت بخشی از هر برنامه C است.
- پرانتزها نشان دهنده این است که main یک تابع است.
- هر برنامه C با اجرای تابع main آغاز می‌شود.
- برنامه اصلی داخل تابع main نوشته می‌شود.

{}

یک بلاک از برنامه، شامل این دو علامت آکولاد باز و بسته و هر آنچه در بین آن دو قرار می‌گیرد، است

خروجی

printf(“welcome to c\n”);

- یک عبارت رشته‌ای را در خروجی چاپ می‌کند.
- عبارت داخل ”“ دقیقا در خروجی چاپ می‌شود.
- \n در خروجی چاپ نمی شود.
- \n یک کاراکتر کنترلی است.
- با دیدن backslash کاراکتر بعدی چک می‌شود.
- بعد از چاپ عبارت، مکان نما به خط بعد می‌رود.

```
1  /* Fig. 2.1: fig02_01.c
2     A first program in C */
3  #include <stdio.h>
4
5  /* function main begins program execution */
6  int main( void )
7  {
8     printf( "Welcome to C!\n" );
9
10    return 0; /* indicate that program ended successfully */
11 } /* end function main */
```

```
Welcome to C!
```

کاراکترهای کنترلی

- کاراکترهای کنترلی با بک اسلش شروع می شوند.

Escape sequence	Description
\n	Newline. Position the cursor at the beginning of the next line.
\t	Horizontal tab. Move the cursor to the next tab stop.
\a	Alert. Sound the system bell.
\\	Backslash. Insert a backslash character in a string.
\"	Double quote. Insert a double-quote character in a string.

- اگر بخواهیم backslash را چاپ کنیم از دوتا پشت‌سرهم استفاده می شود \\
- گیومه دوتایی نیز معنی خاصی دارد بنابراین برای نمایش آن باید از \" استفاده کرد

انجام موفقیت‌آمیز اجرا

- return 0
- آخر هر تابع main قرار می گیرد.
- به معنی خروج از برنامه و انجام موفقیت‌آمیز اجرا است.

```
1  /* Fig. 2.1: fig02_01.c
2     A first program in C */
3  #include <stdio.h>
4
5  /* function main begins program execution */
6  int main( void )
7  {
8     printf( "Welcome to C!\n" );
9
10    return 0; /* indicate that program ended successfully */
11 } /* end function main */
```

ساختار نحوی

- هر دستور با ; نیم دونقطه (سمیکالن) پایان می‌یابد.
- می‌توان در یک خط چند دستور نوشت یا یک دستور طولانی را در چند خط نوشت و نشانه پایان دستور سمیکالن است.

- این زبان به بزرگی و کوچکی حروف حساس است. دستورات با حروف کوچک نوشته می‌شوند.

```
1  /* Fig. 2.1: fig02_01.c
2     A first program in C */
3  #include <stdio.h>
4
5  /* function main begins program execution */
6  int main( void )
7  {
8     printf( "Welcome to C!\n" );
9
10    return 0; /* indicate that program ended successfully */
11 } /* end function main */
```

حالتهای مختلف استفاده از printf

چاپ چند خط خروجی با یک دستور printf

```
1  /* Fig. 2.4: fig02_04.c
2     Printing multiple lines with a single printf */
3  #include <stdio.h>
4
5  /* function main begins program execution */
6  int main( void )
7  {
8     printf( "Welcome\nto\nC!\n" );
9
10    return 0; /* indicate that program ended successfully */
11 } /* end function main */
```

```
Welcome
to
C!
```

حالت‌های مختلف استفاده از printf

چاپ یک خط خروجی با چند دستور printf

```
1  /* Fig. 2.3: fig02_03.c
2     Printing on one line with two printf statements */
3  #include <stdio.h>
4
5  /* function main begins program execution */
6  int main( void )
7  {
8     printf( "Welcome " );
9     printf( "to C!\n" );
10
11    return 0; /* indicate that program ended successfully */
12 } /* end function main */
```

```
Welcome to C!
```

ورودی

خواندن مقدار از کاربر با استفاده از دستور scanf

scanf دو آرگومان دارد

scanf(“بخش اول”, &بخش دوم);

- بخش اول نوع داده‌ای که قرار است خوانده شود را مشخص می کند
- %d یعنی عدد صحیح را بخواند.
- % کاراکتر فرمت است .
- بخش دوم با علامت اند یا امپرسند(&) شروع می شود که آدرس متغیری است که باید از ورودی خوانده شود و بعد از آن باید نام متغیر قرار بگیرد.

برنامه حاصل جمع دو عدد

برنامه ای بنویسید که دو عدد را از ورودی خوانده و حاصل جمع آنها را محاسبه کرده و در خروجی چاپ کند.

```
1  /* Fig. 2.5: fig02_05.c
2     Addition program */
3  #include <stdio.h>
4
5  /* function main begins program execution */
6  int main( void )
7  {
8     int integer1; /* first number to be input by user  */
9     int integer2; /* second number to be input by user */
10    int sum; /* variable in which sum will be stored */
11
12    printf( "Enter first integer\n" ); /* prompt */
13    scanf( "%d", &integer1 ); /* read an integer */
14
15    printf( "Enter second integer\n" ); /* prompt */
16    scanf( "%d", &integer2 ); /* read an integer */
17
18    sum = integer1 + integer2; /* assign total to sum */
19
20    printf( "Sum is %d\n", sum ); /* print sum */
21
22    return 0; /* indicate that program ended successfully */
23 } /* end function main */
```

```
Enter first integer
45
Enter second integer
72
Sum is 117
```

متغیر

```
int integer1;
int integer2;
int sum;
```

نام **متغیرها** هستند `integer1, integer2, sum`

- متغیر، مکانی در حافظه است که مقداری را در خود نگه می‌دارد.
- این متغیرها از نوع int هستند یعنی مقادیر صحیح را نگهداری می‌کنند (7و-11و0و 31914)
- متغیرها باید قبل از استفاده (ترجیحا در ابتدای برنامه) تعریف شوند تا محلی برای آنها در حافظه اختصاص داده شود.

- به صورت زیر هم می‌توان آنها را تعریف کرد

```
int integer1, integer2, sum;
```

متغیر

- نام هر متغیر در زبان C باید یک **شناسه معتبر** باشد.
- شناسه یک مجموعه از حروف، اعداد و آندرلاین می باشد و نباید با رقم شروع شود.
- می تواند به هر طولی باشد ولی فقط 31 کاراکتر اول آن استفاده می شود.
- زبان C حساس به متن است، بنابراین A1 و a1 دو شناسه متفاوت هستند .
- کلمات کلیدی و دستورات نباید به عنوان متغیر تعریف شوند.
- متغیرها دقیقا به همان صورتی که تعریف شده‌اند باید استفاده شوند.

عملگرها

```
scanf( "%d", &integer1 ); /* read an integer */
```

- این دستور مقداری را از کاربر می گیرد.
- از ورودی استاندارد که معمولا کی بورد است مقدار را می خواند.

بعد از اجرای دستور scanf، برنامه منتظر کاربر می ماند تا کاربر عددی را وارد کند سپس عدد وارد شده را به متغیر integer1 نسبت می دهد. از این به بعد مقدار integer1، مقدار وارد شده خواهد بود.

```
scanf( "%d", &integer2 ); /* read an integer */
```

- این دستور مقدار را برای متغیر دوم دریافت می کند.

انتساب

```
sum = integer1 + integer2; /* assign total to sum */
```

- جمع دو متغیر integer1 و integer2 را حساب کرده و حاصل را به sum نسبت می‌دهد.
- عبارت بالا به این صورت خوانده می‌شود:
- متغیر sum حاصل جمع دو متغیر integer1 و integer2 را می‌گیرد.

دستور printf

```
printf( "sum is %d\n", sum ); /* print sum */
```

- تابع printf جمع دو مقدار را چاپ می‌کند و دو تا آرگومان دارد.
  - "sum is %d\n"
  - sum
- بخش اول، رشته کنترل فرمت است.
- %d نشان می‌دهد که یک عدد صحیح چاپ خواهد شد.
- آرگومان دوم، مقداری را مشخص می کند که می خواهد چاپ شود.

می توان دو دستور محاسبه و چاپ خروجی را با هم ترکیب کرد.

```
printf( "sum is %d\n", integer1 + integer2 );
```

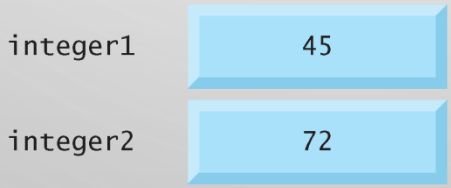
کارکرد حافظه

- نامهای integer1 , integer2 و sum در واقع به مکان‌هایی از حافظه کامپیوتر اشاره دارد.
- هر متغیر داری نوع و مقدار می باشد.

```
scanf( "%d", &integer1 ); /* read an integer */
```

- بعد از اجرای این دستور، مقداری که بوسیله کاربر وارد شده در محلی از حافظه قرار می گیرد که نام آن integer1 است. برای integer2 هم به طور مشابه مقداری در حافظه قرار می گیرد

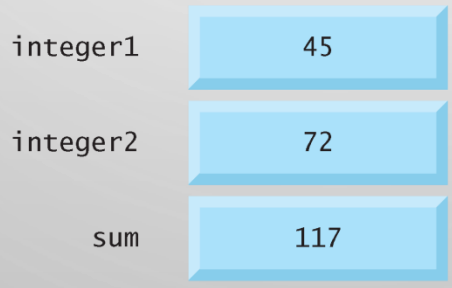
```
scanf( "%d", &integer2 ); /* read an integer */
```



کارکرد حافظه

- وقتی یک مقدار در حافظه قرار می‌گیرد، مقدار قبلی آن از بین می رود .
- با اجرای دستور sum، حاصل‌جمع آنها در محل حافظه sum قرار می گیرد

```
12      printf( "Enter first integer\n" ); /* prompt */
13      scanf( "%d", &integer1 ); /* read an integer */
14
15      printf( "Enter second integer\n" ); /* prompt */
16      scanf( "%d", &integer2 ); /* read an integer */
17
18      sum = integer1 + integer2; /* assign total to sum */
```



عملیات ریاضی

- محاسبات ریاضی در زبان C

C operation	Arithmetic operator	Algebraic expression	C expression
Addition	+	$f + 7$	`f + 7`
Subtraction	–	$p - c$	`p - c`
Multiplication	*	bm	`b * m`
Division	/	x / y or $\frac{x}{y}$ or $x \div y$	`x / y`
Remainder	%	$r \bmod s$	`r % s`

تمرین

- برنامه‌ای بنویسید تا موارد زیر را انجام دهد
  - پیامی نمایش دهد که قرار است حاصل جمع سه عدد را محاسبه کند
  - متغیرهای **x,y,z** و نتیجه از نوع صحیح باشند
  - به کاربر اعلام کند که سه عدد را وارد کند
  - سه عدد را خوانده و در متغیرهای **x,y,z** ذخیره کند
  - حاصل را به متغیری به نام **result** نسبت دهد
  - حاصلجمع را با عبارت «**result is** » در خروجی چاپ کند

تمرین

- برنامه ای بنویسید که طول و عرض مستطیل را از ورودی گرفته و مساحت آن را محاسبه کند.(طول و عرض مستطیل از نوع داده صحیح هستند)