



بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

شبهه کد
ورودی و خروجی

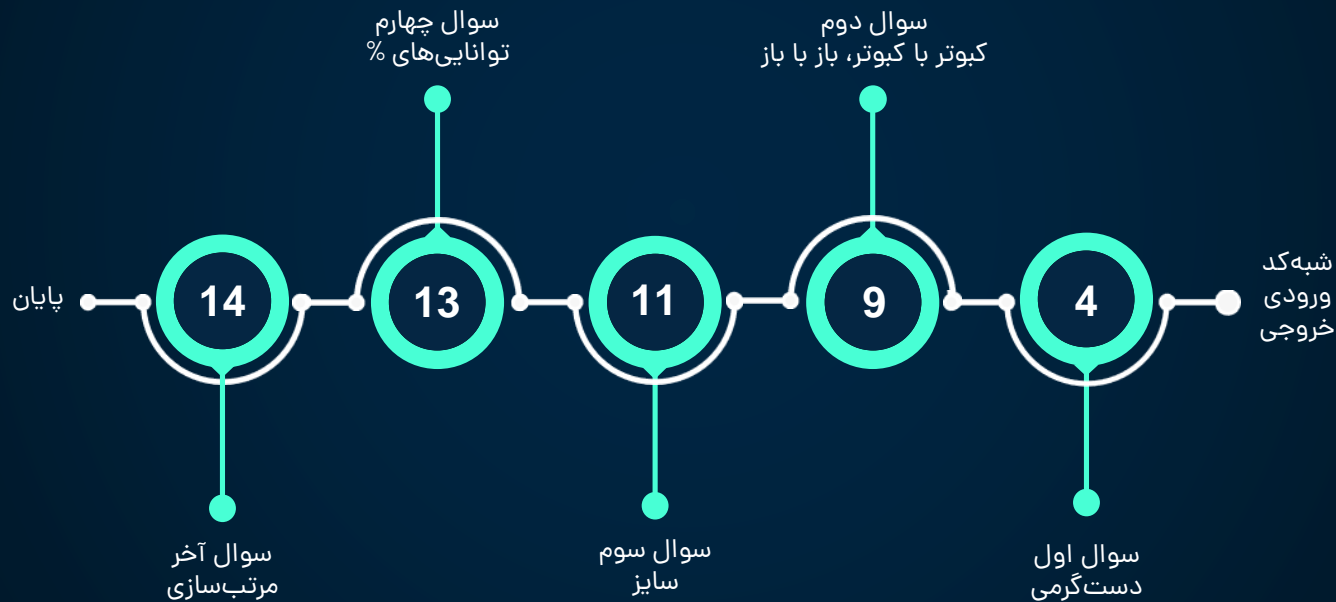
جلسه سوم

برای قدم گذاشتن در دنیای برنامه‌نویسی لازم است تا بتوانیم با کامپیوتر ارتباط برقرار کنیم. در این مسیر باید بدانیم چطور می‌توانیم اطلاعات خود را در کامپیوتر وارد و ذخیره کنیم و همچنین چطور باید از کامپیوتر بخواهیم تا نتیجه‌ی کار خود را به ما نشان دهد.

به همین دلیل این جلسه از کارگاه را به گرفتن ورودی از کاربر و نشان دادن خروجی توسط برنامه اختصاص داده‌ایم.



فهرست



سوال اول: دست‌گرمی

برای باز کردن در ورودی کامپیوتر باید ابزار مناسب این کار را داشته باشید. اولین ابزاری که برای این کار با آن آشنا می‌شوید، تابع `scanf` است.

```
scanf("%type", &variable);
```

ابزار مورد نیاز بعدی ما تابع `printf` هست که به برنامه‌ی ما این امکان را می‌دهد تا بتواند خروجی رو نشان دهد.

```
printf("%type", variable);
```

هر دوی این ابزارها از داخل جعبه ابزاری با نام `stdio` مخفف `standard input output` آمده‌اند. پس باید حتما در ابتدای برنامه نام این جعبه ابزار (که در دنیای کامپیوتر و برنامه‌نویسی به جای جعبه ابزار به آن کتابخانه گفته می‌شود) آورده شود.

هر چیزی که بخواد از طریق صفحه کلید وارد کامپیوتر بشه، باید از دروازه‌ای که تو شکل بالا سبز رنگ کشیده شده عبور کنه. اگر قرار باشه در زمان اجرای برنامه، ورودی‌ای از دنیای خارج به برنامه داده بشه، باید مشخصاتش رو به دروازه بدیم تا اجازه‌ی ورودش داده بشه.

`scanf(`  `, &x);`

`printf(`  `, x);`

بعد هم یک راهنما براش تعیین کنیم تا بتونه با کمک اون بره خونه‌ی خودش توی حافظه. اگر این راهنما نباشه ممکنه ورودی ما توی کامپیوتر سرگردان بمونه و جاش رو پیدا نکنه!!
اما برای چاپ کردنش هرچند باید دوباره از همون دروازه‌ی سبزرنگ عبور کنه اما دیگه به راهنمایی احتیاج نداره چون خودش راه رو یاد گرفته :)))



همانطور که گفته شد باید زمانی که متغیری به این توابع داده می‌شود، مشخصاتش هم ذکر شود. مشخصات مختلفی که یک متغیر می‌تواند داشته باشد، در جدول زیر نوشته شده و در ستون روبه‌روی آن‌ها علامت معادلش در زبان C نوشته شده.

Format or Type	Format Specifier
int	%d or %i
char	%c
float	%f
double	%lf



بنابراین اگر قرار باشد یک عدد صحیح مثل x وارد کامپیوتر شود باید گفته شود که نوع این متغیر integer یا همان صحیح است و به این صورت گفته می‌شود:

```
scanf("%d", &x);
```

حال قطعه کد زیر را اجرا کنید تا توضیحات قبلی بیش‌تر برایتان جا بیفتد. نام کتابخانه‌ی مورد نیاز هم همانطور که می‌بینید در ابتدای کد آمده است.


```
#include <stdio.h>
```


```
int main() {  
    int x, y;  
    scanf("%d", &x);  
    scanf("%d", &y);  
    printf("The result is: %d\n", (x + y));  
}
```

به نظر شما هر تغییری در کد منجر به اتفاقی خواهد شد؟

سوال دوم: کبوتر با کبوتر، باز با باز

کامپیوترها همه چیز را به صورت اعداد 0 و 1 ای ذخیره می کنند، اما این رشته های 0 و 1 را به انواع مختلفی تفسیر می کنند و دقیقا به همین دلیل است که ما باید در هنگام استفاده از متغیرها، نوع تفسیر (به صورت دقیق تر، type یا نوع) آن متغیر را نیز بدانیم. در غیر این صورت، کامپیوتر تفسیر نادرستی از متغیر 0 و 1 ای ما خواهد داشت و نمی توانیم محاسبات درستی انجام دهیم.


مثلا اگر به کامپیوتر بگوییم که x یک عدد صحیح است اما هنگام ورودی دادن به برنامه عدد 12.7 را وارد کنیم چه اتفاقی خواهد افتاد؟ برعکس آن چطور؟ یعنی اگر از کامپیوتر بخواهیم عدد 56 را به صورت integer یعنی با `%d` چاپ کند، مشکلی به وجود خواهد آمد؟ 

برای بررسی بیش‌تر کد زیر را ببینید. خروجی این کد چیست؟ می‌توانید قبل از اجرای برنامه و با کمک کد، آن را تفسیر کنید؟ 

این خروجی در اصل چه چیزی که مربوط به ورودی شما باشد را نشان می‌دهد؟
به ازای کاراکترهای مختلفی که وارد می‌کنید خروجی‌ها چه تغییری می‌کنند؟

```
#include <stdio.h>
```

```
int main() {  
    char x;  
    printf("Enter a character:\n");  
    scanf("%c", &x);  
    printf("%d\n", x);  
}
```

می‌توانید شباهت‌های ورودی و خروجی خود را با محتوای جدول موجود در لینک زیر مقایسه کنید؟ 



ASCII lookup table

<https://b2n.ir/202623>



این بار قطعه کد زیر را اجرا کنید. سپس به عنوان ورودی یکبار عدد 5 و یکبار عدد 1092091904 را وارد کنید.

```
#include <stdio.h>

int main() {
    float x;
    printf("Enter a decimal number:\n");
    scanf("%d", &x);
    printf("%f\n", x);
}
```



عدد مشاهده شده در خروجی نمایانگر چه مقداری است؟ مقدار عدد مشاهده شده در خروجی به ازای ورودی 1092091904 را در سیستم Floating Point محاسبه کنید. سپس با استفاده از مبدل اعداد binary به decimal در لینک زیر مقدار عدد را در مبنای 10 محاسبه کنید. چه نتیجه‌ای می‌گیرید؟

سوال سوم: سائز



هر کدام از متغیرها در کامپیوتر سائز خاصی دارند یعنی تعداد بایت^۱ مشخصی را قرار است از حافظه‌ی کامپیوتر اشغال کنند.

نکته‌ی بی‌ربط: هر بایت برابر هشت بیت است و هر بیت نماینده‌ی یک 0 یا 1 در کامپیوتر می‌باشد.

```
#include <stdio.h>
```

با اجرای کد زیر سائز هر کدام از متغیرها را به دست آورید.

```
int main() {  
    printf("int: %ld \n", sizeof(int));  
    printf("float: %ld \n", sizeof(float));  
    printf("double: %ld \n", sizeof(double));  
    printf("char: %ld \n", sizeof(char));  
}
```

اگر به جای int از long استفاده کنیم چه فرقی می‌کند؟

long long int چگونه؟



قطعه کد زیر را با ورودی 1399 امتحان کنید. چه اتفاقی می افتد؟

```
#include <stdio.h>
```

```
int main() {  
    char x;  
    printf("Enter a number:\n");  
    scanf("%d", &x);  
    printf("%c\n", x);  
}
```



آیا برنامه با خطایی مواجه شد؟

می توانید دلیل این خطا را با توجه به نکته ای که

از کد قبلی به دست آورده اید توجیه کنید؟

در صورتی که برنامه ی شما با خطایی مواجه نشده است، علت آن هوشمندی کامپایلری است که از آن استفاده می کنید. یعنی خودش متوجه شده که متغیر شما از نوع char یا کاراکتر است و عدد ورودی را به عنوان کد اسکی آن در نظر گرفته و خودش کاراکتر متناظر با آن کد اسکی را در حافظه ذخیره می کند.

سوال چهارم: توانایی‌های %

این مشخص‌کننده‌های قالبی^۱ که تا الان دیدید، علاوه بر فهماندن نوع متغیر، ابزارهای دیگری هم دارند که به صورت کلی به این شکل نمایش داده می‌شود.



`%[-][width].[precision][conversion character]`



برای تمرین هر کدام از این بخش‌ها یک برنامه بنویسید که اعداد زیر را به عنوان ورودی دریافت کند و آن‌ها را طوری چاپ کند که خروجی شبیه به یک جدول با دو سطر و سه ستون باشد. تمامی ستون‌ها به اندازه‌ی ۶ کاراکتر فضا داشته باشند و اعضای ستون دوم همگی تا ۳ رقم اعشار را نشان دهند.



0.00 و 12 و 67831 و 779892.2422 و 3 و 33.0



و اما سوال آخر: مرتب سازی



سلام بچه ها :) ورودتون یا شایدم ورود دوباره تون رو به برنامه نویسی تبریک می گم. فکر کنم دیگه تا الان حسابی توی سوالات دستورکار با ورودی و خروجی کار کردین و با نکاتش تا حد خوبی آشنا شدین. برای همین به نظرم بهتره که الان برگردیم سراغ شبهه کد تا با هم دوباره برخی از نکاتی که توی الگوریتمها لازمه رعایت بشه رو بررسی کنیم.



اول از همه من هم سلام کنم و دوم از همه (!) در تکمیل صحبت های کدخدا بگم که اهمیت الگوریتم پیدا کردن و اینکه چطور باید حرفمون رو به کامپیوتر بفهمونیم، خیلی از اینکه چطور کدش رو بزنیم مهم تره. برای تمرین این توانایی می تونین تو هر پدیده یا اتفاق اطرافتون بگردین و تصور کنین اون رو قراره برای یک آدم که همزبان شما نیست، فقط و فقط در قالب کلمات، توضیح بدین.



مثلا یکی از این پدیده‌ها چی می‌تونه باشه؟

فرض کنیم برای حل مسئله‌ای نیاز به مرتب کردن یه سری عدد داریم. اول از همه باید ببینیم ما خودمون چطور این کار رو انجام می‌دیم تا بتونیم راه‌حلمون رو مرحله به مرحله برای این دوست غیر هم‌زبانمون توضیح بدیم.



خب پس اول بیاین یه مثال بزنیم برای خودمون... ذهن ما برحسب عادت چطور این رشته از عدد رو مرتب می‌کنه؟ هر راه حل و روندی که تو ذهنتونه رو بگین.

{3, 11, 5, 2, 17, 4, 1}

راه‌های مختلفی هست. حالا باید یه طوری عمل کنیم که این روند رو با کمک الگوریتم‌ها برای کامپیوتر هم قابل درک کنیم.



یه راه می‌تونه چی باشه؟ اینکه اول بگردیم کوچک‌ترین عدد توی این رشته از اعداد کدومه و اون رو بذاریم اول صف. بعد بریم توی بقیه ببینیم کوچک‌ترین کدومه و بره جای نفر دوم و ... همین‌طوری تا آخر ادامه بدیم تا جای که همه‌ی عدد هامون مرتب شده باشن.

به این الگوریتم می‌گن **selection sort**.



هر کدوم از لینک‌های زیر برای توضیح همین الگوریتم هستن. در صفحه‌ی بعد هم می‌تونین روند کلی الگوریتم رو ببینین.



Selection Sort Algorithm

<https://b2n.ir/600734>



Selection Sort Algorithm Visualizer

<https://b2n.ir/489622>



Select-sort with Gypsy folk dance

<https://b2n.ir/646876>



سعی کنین با هم گروهی تون شبه کد این الگوریتم رو طوری بنویسین که برای کامپیوتر قابل درک باشه.

- 1- Find the smallest card. Swap it with the first card.
- 2- Find the second-smallest card. Swap it with the second card.
- 3- Find the third-smallest card. Swap it with the third card.
- 4- Repeat finding the next-smallest card, and swapping it into the correct position until the array is sorted.



کسی می تونه بگه منظور از قابل درک بودن چی هست؟
مثلا اگه به یک آدمی که زبان شما رو متوجه نمی شه بگین کوچک ترین عدد بین این اعداد رو پیدا کن، متوجه حرفتون می شه؟
پس چطور باید منظورمون رو بهش بفهمونیم؟



الگوریتم دیگه‌ای که می‌خواهیم باهاش آشنا بشیم، مرتب‌سازی حبابی یا **bubble sort** هست. این الگوریتم یکی از الگوریتم‌های مهم محسوب می‌شه که باید حسابی روش تسلط داشته باشیم. زیاد به کارتون میاد: دی



این یکی رو سعی کنین از طریق لینک‌های زیر متوجه روندش بشین و بعد با کمک هم‌گروهی‌تون شبه‌کدش رو بنویسین.



Bubble Sort Algorithm Visualizer

<https://b2n.ir/218225>



Bubble-sort with folk dance

<https://b2n.ir/294191>



الگوریتم‌های مرتب‌سازی خیلی زیادن و محدود به همین دو مورد نیستن. اگر دوست داشتن الگوریتم‌های بیش‌تری رو ببینین با یه سرچ کوچولو به راحتی بقیه‌شون هم پیدا می‌شن. البته توی همون لینک‌هایی که برای selection sort و bubble sort بود هم می‌تونین دنبال بقیه‌شون بگردین؛



امیدوارم که از این کارگاه لذت برده باشین و هم‌صحبتی با یه موجود زبون‌نفهم (البته خیلی باهوش که به موقعش خیلی خوب می‌فهمه و درس پس می‌ده) براتون دل‌نشین بوده باشه. تا جلسه‌ی بعد خدانگه‌دار...

;