تابع بازگشتی ۲

مبانی برنامه‌نویسی

(۱۳۹۹-۱۳-۱۱)

جلسه‌ی بیست‌ودوم

دانشگاه شهید بهشتی

پاییز ۱۳۹۲

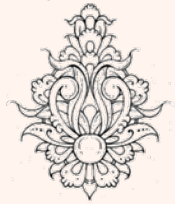
دانشکده‌ی مهندسی برق و کامپیوتر

احمد محمودی ازناوه

http://faculties.sbu.ac.ir/~a_mahmoudi/

فهرست مطالب

- مروری بر جلسه‌ی پیش
- آشنایی توابع بازگشتی
- حل مسأله به صورت بازگشتی



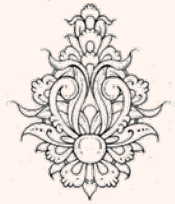
- ویرایش جدید با کمک آقای «محمدرضا بهرامی» تهیه شده است.

تمرین

- خروجی برنامه‌ی زیر چیست؟

```
void showMe(int arg);
int main()
{
     int num = 0;
     showMe(num);
     return 0;
}
void showMe(int arg)
{
     if (arg < 10)
          showMe(++arg);
     cout << arg << endl;
}
```

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

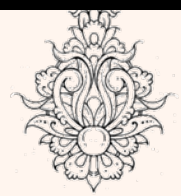


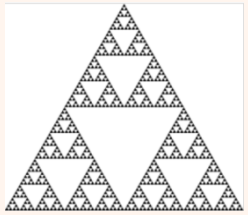
تمرین

```
void printnum ( int begin )
{
   cout<< begin;
   if ( begin < 9 )
   {
       printnum ( begin + 1 );
   }
   cout<< begin;

}

int main()
{
   printnum ( 4 );
}
```

- خروجی چیست؟

```
456789987654
```





الگوریتم‌های تکراری

- برای پیاده‌سازیِ الگوریتم‌هایی که فرایندی را تکرار می‌کنند معمولاً به دو شیوه عمل می‌شود:

استفاده از حلقه‌ها

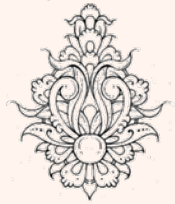
- توابع بازگشتی

استفاده از توابع بازگشتی

  - برنامه‌نویس مستقیما فکر خود را پیاده‌سازی کرده و درگیر مسائل جانبی از قبیل کنترل حلقه و متغیرها نمی‌شود.
  - پیاده‌سازی با دستورالعمل‌های کم‌تر همراه با ظرافت و خوانایی بهتر برنامه

توابع بازگشتی

- در اینجا قصد داریم «**توابع بازگشتی**» را به عنوان ابزاری برای حل دسته‌ای وسیع از مسائل معرفی کنیم. این ابزار قدرتمند حل بسیاری از مسائل را ممکن کرده یا پیاده‌سازی کد را بسیار ساده می‌کند.
- به جهت اهمیّت این مبحث و دخیل بودن مبانی ریاضی، کمی از برنامه‌نویسی فاصله گرفته و به بحث پیرامون مطالب زیر می‌پردازیم:
  - **استقرای ریاضی**
  - **روابط بازگشتی**
  - **طراحی توابع بازگشتی** (حل با استفاده از ساختار بازگشتی)



استقرای ریاضی

- همه شما در دبیرستان با استقرای ریاضی آشنا شده‌اید و از آن برای اثبات برخی از احکام ریاضی استفاده کرده‌اید.
- به طور مثال:
- ***ثابت کنید که هر عدد طبیعی بزرگ‌تر از « ۱ » را می‌توان به عوامل اول تجزیه کرد.***
- همان‌طور که به یاد دارید برای حل یک مسئله توسط استقرا به یک پایه و یک فرض نیازمندیم. سپس با استفاده از گام استقرایی و نتیجه گرفتن حکم برای یک عدد طبیعی خارج از مجموعه فرض، حل را کامل می‌کردیم. یعنی از پایه بدیهی حکم را برای بقیه اعداد تعمیم می‌دادیم.

حل

- **پایه‌ی استقرا:**

بدیهی است که در حالت $n = 2$ همه چیز صحیح است.

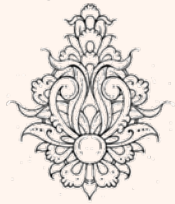
- **فرض قوی استقرا:**

فرض می‌کنیم برای تمام اعداد کوچک‌تر از k مسئله درست است.

- **گام استقرایی:**

عدد $k+1$ یا اول است یا مرکب :

اگر اول باشد: چون عدد اول است می‌توان آن را به صورت $1*(k+1)$ نوشت و اثبات تمام است.



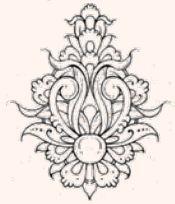
حل (ادامه...)

- اگر مرکب باشد:

چون این عدد مرکب است پس حتما عددی مانند p که اولا، اول است و ثانیا از $k+1$ کوچک‌تر است وجود دارد به طوری که:

$$k+1 = p*q$$

که q عددی کوچک‌تر از $k+1$ است و طبق فرض قوی استقرا می‌توان q را به صورت حاصل ضرب اعداد اول نوشت و در اینجا کار تمام است. ■



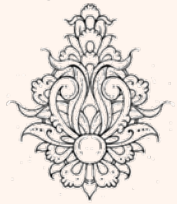
آشنایی با روابط بازگشتی

- گاهی قادر به حل مسائل به خصوص مسائل شمارشی به صورت صریح نیستیم ولی در صورت داشتن جواب مسئله در حالات دیگر به راحتی قادر به حل سوال در حالت جدید خواهیم بود.
- به مثال زیر که یک مسئله‌ی شمارشی است توجه کنید:

- ***۱۰ سکه در اختیار دارید که هر یک داری دو طرف سفید و سیاه هستند، با این فرض که هیچ دو سکه سیاهی کنار هم نباشند. تعداد حالت های ممکن چینش را حساب کنید؟***

روابط بازگشتی (مثال)

- اگر به صورت‌بندی مسئله بیشتر دقت کنیم می‌توانیم بهتر آن را درک کنیم.
- اگر کمی بیشتر به مسئله فکر کنیم درخواهیم یافت که مجموعه جواب‌های مسئله به دو دسته افراز می‌شوند:
  - دسته‌ی اول شامل اعضایی هستند که اولین سکه در آن‌ها به رنگ سفید است.
  - دسته دوم شامل آن‌هایی که با سیاه آغاز می‌شوند.
- بدیهی است که عناصر دوم از دسته دوم حتما به رنگ سفید هستند (**چرا؟**)
- قبل از ادامه‌ی داستان(!) توجه شما را به این حالات بدیهی جلب می‌کنیم:



مثال (ادامه...)

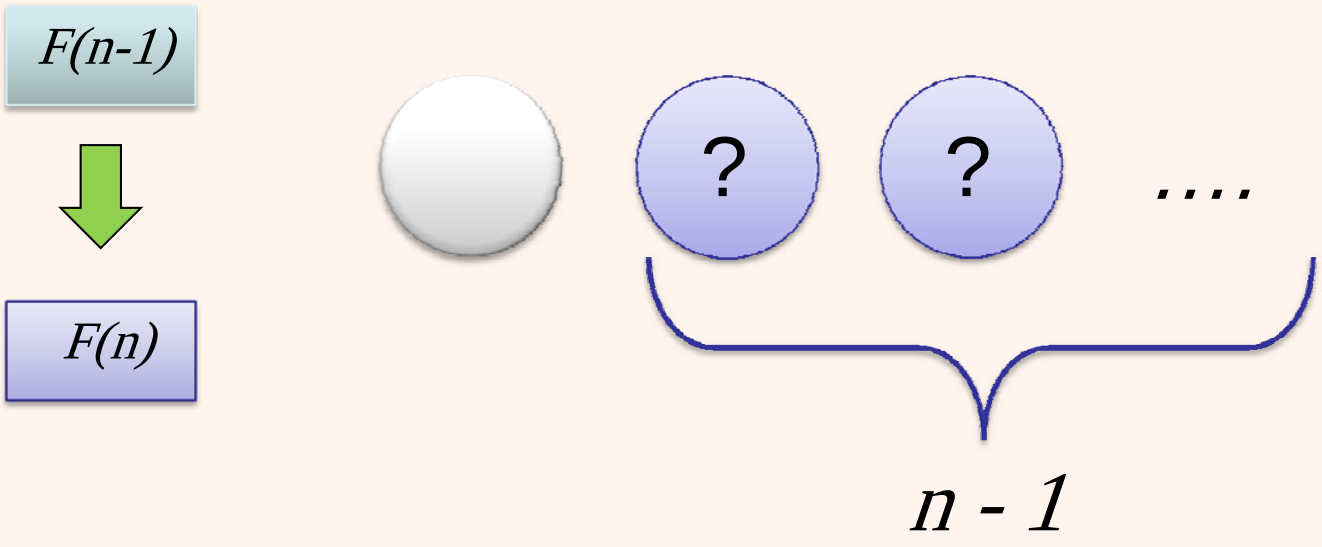
- فرض کنید که تعداد جواب‌ها برای تعداد n سکه با F(n) نشان داده شوند. در این صورت:

- F(1) = 2 
- F(2) = 3 
- ....

- بیایید کمی به عقب برگردیم، به استقرای ریاضی!
- مانند استقرا فرض کنید حل مسئله برای تمام اعداد کوچک‌تر از n ممکن بوده و جواب را داریم!

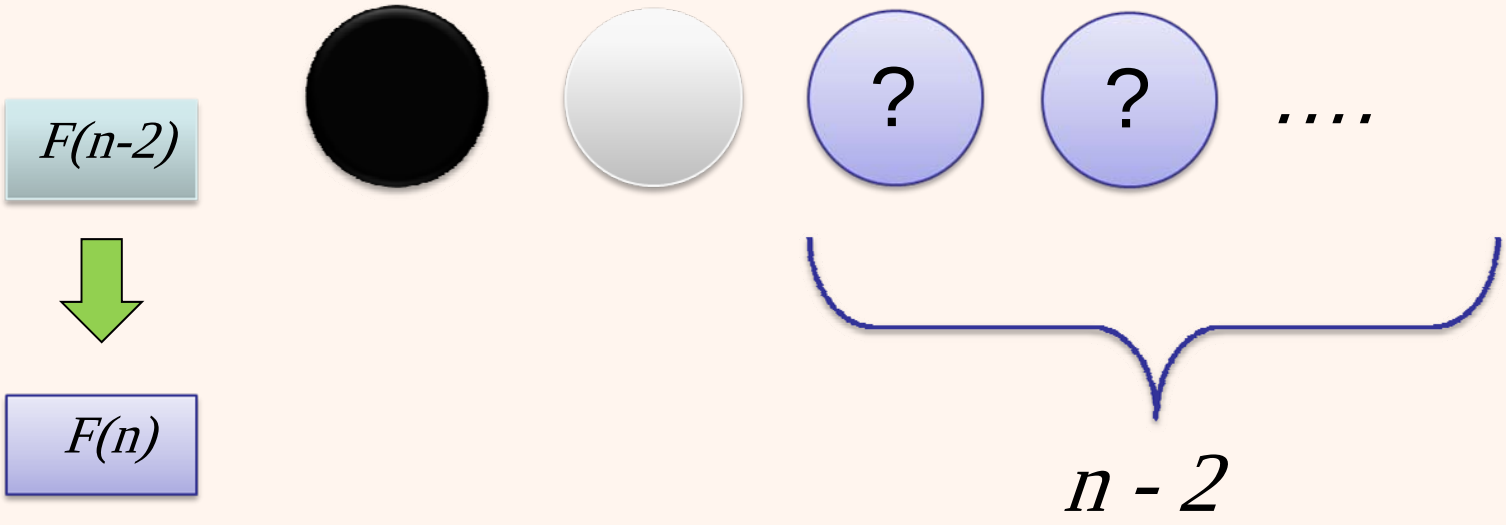
مثال (ادامه...)

- آیا می‌شود جواب را محاسبه کرد؟
- گفتیم جواب‌ها چگونه افراز خواهند شد؛ اگر حالت اول را در نظر بگیرید با داشتن F(n-1) همه جواب‌ها را خواهید داشت:



مثال (ادامه...)

- اما از طرفی دیدیم که در حالت دوم مجبور به گذاشتن سفید در خانه دوم هستیم پس:
  - اگر حاصل F(n-2) را داشته باشیم قادر به محاسبه این مجموع نیز هستیم(افزودن تصاویر):

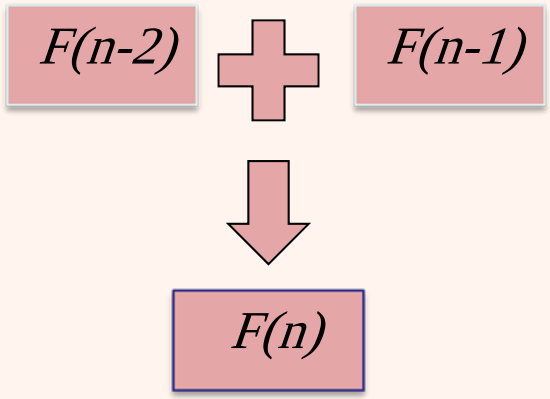


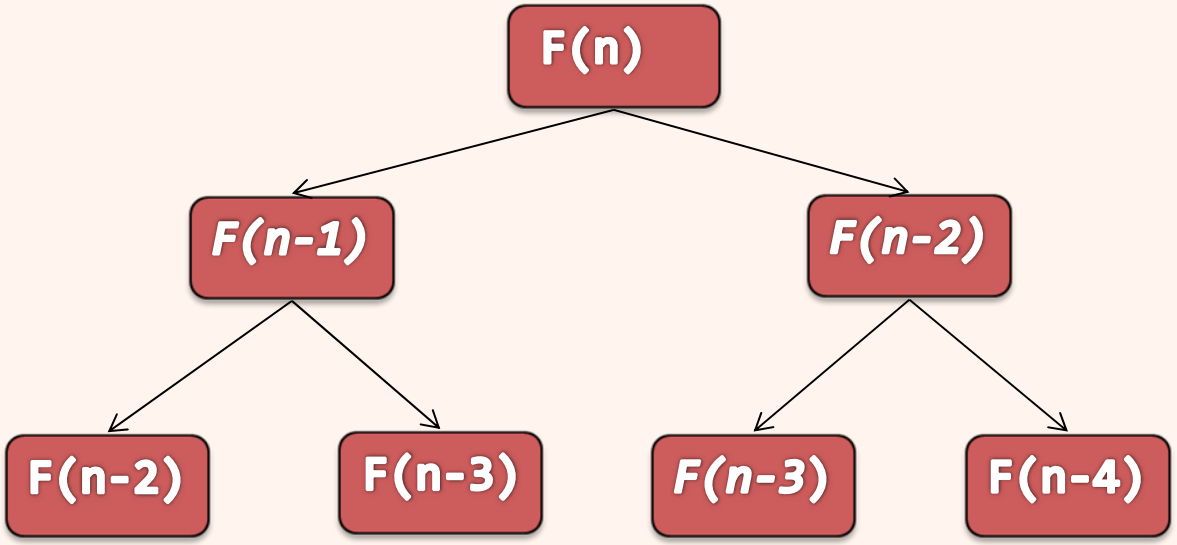
مثال (ادامه...)

- بنابراین برای جوابمان خواهیم داشت:

$$F(n) = F(n-1) + F(n-2)$$

- در اینجا گام استقرایی ما برای رسیدن به جواب یک عمل جمع است!





مثال (ادامه...)

- پس جواب خواهد شد:

- F(1) = 2
- F(2) = 3
- F(3) = 5
- F(4) = 8
- F(5) = 13
- F(4) = 21
- F(7) = 34
- F(8) = 55
- F(9) = 89

F(10) = 144

حل مسئله به کمک توابع بازگشتی

- در اینجا می خواهیم با طراحی الگوریتم به روش استقرایی آشنا شویم.
- استقرا مانند دومینو عمل می‌کند، می‌خواهیم با انداختن اولی دومینو بقیه را نیز واژگون سازیم.
- تنها کافی است مسئله را برای نمونه‌های کوچک حل کرده و سپس، از افتادن هر دومینو، واژگونی دومینو جلویی نتیجه گرفته شود.

- **پس برای حل مسئله باید نشان داد که:**

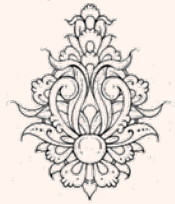
۱- برای نمونه کوچکی قابل حل است. (حالت پایه)

معمولا برای n= 0 یا n = 1 این مورد به راحتی قابل بررسی است.

۲- می‌توان راه حل نمونه بزرگ‌تر را از روی نمونه‌های کوچک‌تر یافت.(گام استقرا).

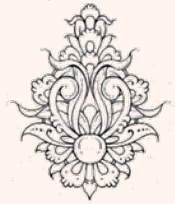
با فرض صحیح بودن جمله(یا جمله‌های) کوچک‌تر و به دست آوردن رابطه برای جمله بزرگ‌تر.

$$F(N) = G(F(n-1), F(n-2), \ldots, F(n-k))$$



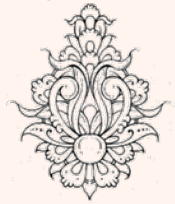
حل و پیاده‌سازی

- ابتدا سعی کنید یک رابطه بازگشتی برای مسئله، مبتی بر جملات کوچک‌تری که فکر می‌کنید دخیل هستند، بنویسید.
- سپس درستی را برای شروط پایه را بررسی کنید.
- رابطه بازگشتی خود را ثابت کنید.
- **توجه:** *اگر ایده‌ای ندارید شروع به حل برای حالات کوچک مسئله و بررسی مثال نمایید.*



حل و پیاده‌سازی(ادامه...)

- **در پیاده‌سازی:**
  - پارامترهای ورودی را بر اساس استقرا و نیاز به حافظه مشخص کنید.
  - ابتدا شرط پایه را بررسی کرده و جواب آن را مشخص کنید.
  - سپس گام استقرایی را با فرض اینکه ***تابع شما درست کار می کند***، بردارید. یعنی اگر نیاز به جواب‌های کوچک‌تر دارید از تابعی که در حال نوشتن آن هستید استفاده کرده و فرض کنید درست کار می‌کند.



مثال

- اعداد فیبوناچی:

یک سری از اعداد که هر جمله آن برابر جمع دو جمله قبل است مشروط بر آنکه جمله اول آن، «صفر» و جمله دوم آن «یک» باشد. یعنی:

$$F(0) = 0$$

$$F(1) = 1$$

حال می‌خواهیم nامین جمله این سری را با استفاده از تعریف، حساب کنیم.

- ۱- طراحی روابط بازگشتی:

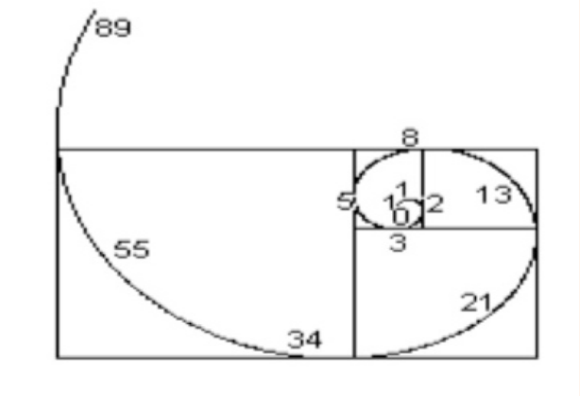
همان‌طور که گفته شد این رابطه برابر با:

$$F(n) = F(n-1) + F(n-2)$$

۲- بررسی درستی:

با توجه به تعریف صحیح است.

0, 1, 1, 2, 3, 5, 8, 13, 21, 34 ,....



پاسخ

```
long fibonacci(unsigned long);

int main()
{   long num;
    cout << "Enter a positive integer: ";
    cin >> num;
    cout << "Fibonacci numbers: " << fibonacci(num) << endl;

}
long fibonacci(unsigned long n)
{
        if (n <= 1) {
              return n;
           }
        else
        {
             return fibonacci(n-1)+fibonacci(n-2);
        }
}
```

گام اول : شرط پایه

گام دوم : گام استقراء

```
Enter a positive integer: 7
Fibonacci numbers: 13
```

پاسخ(ادامه...)

- نحوه‌ی محاسبه به صورت زیر است: