به نام خدا

جزوه جلسه پنجم درس برنامه نویسی با متلب

1- دنباله فیبوناچی

دنباله فیبوناچی دنباله‌ای است از اعداد صحیح که اولین و دومین عدد آن 1 می‌باشد و اعداد بعدی مجموع دو عدد ماقبل می‌باشند. فرمول آن به شکل زیر است.

$$F_n = F_{n-1} + F_{n-2}, F_1 = 1, F_2 = 1$$

اعداد این دنباله عبارتند از ... 1 1 2 3 5 8 13 21 34 55 89

برنامه زیر 14 عدد از اعداد دنباله فیبوناچی را محاسبه می‌نماید

```
clc
clearvars
F = [1 1];
a = 1; b = 1;
for k=1:12
    t = a + b;
    F = [ F t ];
    a = b;
    b = t;
end
disp(F)
```

(نکته: در تعریف جدید عدد صفر را به عنوان اولین عدد از دنباله فیبوناتچی در نظر می‌گیرند)

مثال شماره 11 : برنامه‌ای بنویسید که 10 عدد از اعداد دنباله زیر را تولید نماید

$$F_{n+1} = 3 \times F_n + 2 \times F_{n-1}, F_1 = 3, F_0 = 1$$

```
clc
clearvars
S = [1 3];
a = 1; b = 3;
for k=1:8
    t = 3*b + 2*a;
    S = [ S t ];
    a = b;
    b = t;
end
disp(S)
```

S

1x10 double

	1	2	3	4	5	6	7	8	9	10
1	1	3	11	39	139	495	1763	6279	22363	79647

2- مثلث پاسکال (خیام،نیوتن)

هرگاه ضرایب چند جمله‌ای $(a+b)^n$ را برای n=0 الی n=k زیر هم بنویسیم مثلث جالبی تولید می‌شود که به آن مثلث پاسکال (خیام، نیوتن) می‌گویند.

```
          1
        1   1
      1   2   1
    1   3   3   1
  1   4   6   4   1
1   5  10  10   5   1
1   6  15  20  15   6   1
```

مثال شماره 12 : به سه روش برنامه‌ای بنویسید که سطر k ام از مثلث پاسکال را تولید نماید. فرض کنید بالاترین سطر k=0 است.

```
clc; clearvars;
N = input('please enter N: ');
P = [1 1];
if(N==0)
    disp(1)
elseif (N==1)
    disp(P)
else
    A = [1];
    for k=2:N
        for h=2:k
            A(h) = P(h-1)+P(h);
        end
        A(h+1) = 1;
        P = A;
    end
    disp(A)
end
```

```
clc
clearvars
N = input('please enter N: ');
A = [];
for k=0:N
    A(k+1) = nchoosek(N,k);
end
disp(A)
```

```
clc
clearvars
N = input('please enter N: ');
A = [];
for k=0:N
    A(k+1) = factorial(N)/(factorial(k)*factorial(N-k));
end
disp(A)
```

3- مقسوم‌علیه‌های مثبت عدد صحیح و مثبت N

برنامه زیر مقسوم‌علیه‌های مثبت عدد صحیح و مثبت N را نشان می‌دهد. راه حل اول الگوریتمیک است و راه حل دوم بر اساس دستور divisors است.

```
clc
clearvars
N = input('please enter N: ');
D = [];
for k=1:N
    if rem(N,k)==0
        D = [D k];
    end
end
disp(D)

%divisors(N)
```

هرگاه مجموع مقسوم‌علیه‌های یک عدد (به استثنای خود آن عدد) برابر با همان عدد شود آن عدد را یک عدد کامل (perfect number) می‌گویند. اگر مجموع مقسوم‌علیه‌های یک عدد (به استثنای خود آن عدد) بزرگتر از آن عدد

شود به آن عدد زائد (abundant number) می‌گویند و اگر مجموع مقسوم‌علیه‌های یک عدد (به استثنای خود آن عدد) کوچکتر از آن عدد شود به آن عدد ناقص (deficient number) می‌گویند.

نکته: اعداد کامل بسیار کمیاب هستند

number	divisors	sum of divisors	
6	1 2 3	6	perfect number
15	1 3 5	9	deficient number
20	1 2 4 5 10	22	abundant number

مثال شماره 13 : به دو روش برنامه‌ای بنویسید که یک عدد را از ورودی دریافت نموده و مشخص کند آیا عدد ناقص، زائد یا کامل است.

```
clc
clearvars
N = input('please enter N: ');
S = 0;
for k=1:N-1
    if rem(N,k)==0
        S = S + k;
    end
end
% S = sum(divisors(N))-N;
if S==N
    disp('perfect number')
elseif S<N
    disp('deficient number')
else
    disp('abundent number')
end
```

4- اعداد اول

عددی اول است که فقط بر خود و بر عدد یک بخشپذیر باشد. برای آنکه چک کنیم عددی اول است یا اول نیست از دستور isprime(N) استفاده می گردد. برای یافتن اعداد اول بین 1 الی N از دستور primes(N) استفاده می‌گردد. روش الگوریتمیک برای چک کردن اول بودن یک عدد آنست که چک کنیم آیا مجموع مقسوم‌علیه‌های آن برابر است با N+1. برای آنکه اعداد اول بین 1 تا N را پیدا کنیم می‌توانیم از روش الگوریتمیک استفاده کنیم.

برای این کار اعداد 1 الی N را درون یک حلقه یکی یکی چک می‌کنیم (حلقه خارجی) نحوه چک کردن به این صورت است که مجموع مقسوم‌علیه‌های هر عدد را محاسبه می‌کنیم هر عدد را بر اعداد 1 تا خودش تقسیم می‌کنیم و هرگاه باقی‌مانده صفر شد آن عدد را به مجموع مقسوم‌علیه‌ها اضافه می‌کنیم تا حلقه داخلی به پایان برسد سپس چک می‌کنیم اگر مجموع مقسوم‌علیه‌ها برابر با خود عدد به اضافه یک باشد آن عدد اول است و آن را درون بردار P اضافه می‌کنیم.

مثال شماره 14 : به دو روش برنامه‌ای بنویسید که اعداد اول بین 1 الی 100 را نمایش دهد.

```
clc
clearvars
N = input('please enter N: ');
P = [];
for k=1:N
    S = 0;
    for h = 1:k
        if rem(k,h)==0
            S = S + h;
        end
    end
    if S==(k+1)
        P = [ P k ];
    end
end
disp(P)
%primes(N)
```

مثال شماره 15 : برنامه‌ای بنویسید که یک عدد صحیح مثبت N را دریافت نماید و سپس کوچکترین عدد اول بزرگتر مساوی با N را نمایش دهد. سپس فاصله این دو عدد را نشان دهد.

```
clc
clearvars
N = input('please enter N: ');
k = N;
while ~isprime(k)
    k = k + 1;
end
disp(k)
disp(k-N)
```