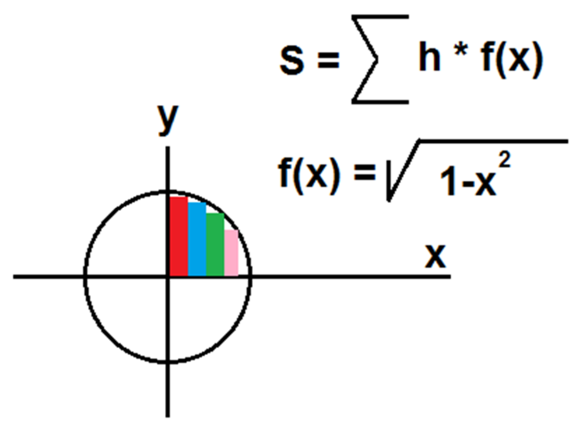
به نام خدا

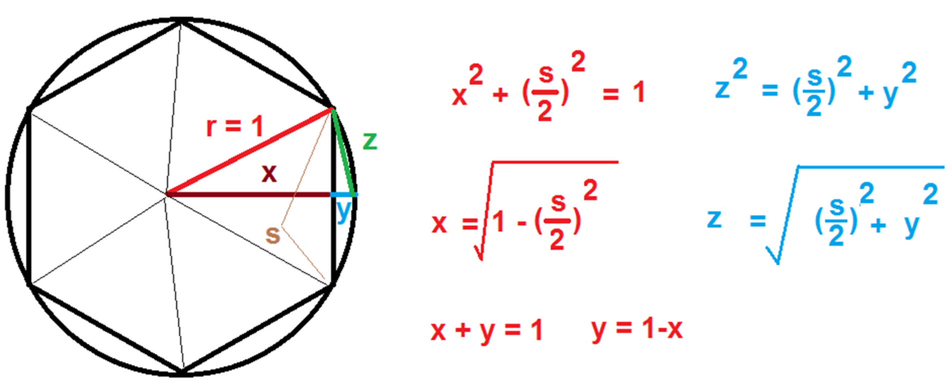
جزوه برنامه نویسی متلب، جلسه هفتم

محاسبه عدد π : برای محاسبه عدد π روش‌های مختلفی وجود دارد. برای شروع می‌توان از مساحت دایره استفاده کرد. برای این منظور ابتدا سعی می کنیم به ساده‌ترین روش مساحت زیر نمودار تابع f(x) را بین ۰ تا ۱ محاسبه نماییم. این تابع نمودار دایره به شعاع واحد در قسمت مثبت محور y است. بنابراین باید مساحت مستطیل‌های زیر نمودار را با هم جمع بزنیم. ابتدا فاصله ۰ تا ۱ را به N قسمت تقسیم می‌کنیم. طول هر قسمت h=1/N می‌باشد که ضلع کوچکتر مستطیل است. ضلع بزرگ آن مقدار تابع در نقطه x است و حاصلضرب اضلاع برابر با مساحت مستطیل می‌باشد لذا باید با استفاده از عمل تکرار این محاسبه را برای N مستطیل انجام دهیم تا مساحت یک چهارم دایره بدست آید سپس آنرا در عدد چهار ضرب می کنیم تا عدد π بدست آید. این روش مناسب نیست زیرا همگرایی آن کم و خطای آن بالا می باشد.

```
%integration in the basic form to calculate PI
clc
clearvars
s = 0;
N = 1000;
delta = 1/N;
x = 0;
for i = 1:N
    s = s + delta*(sqrt(1-x^2));
    x = x + delta;
end
s = 4*s
```



روش دوم نیز بر اساس مثلث بندی دایره انجام می‌گیرد در این روش به جای مساحت از محیط دایره واحد استفاده می‌گردد.



محیط دایره واحد برابر است با 2π برای محاسبه در گام نخست دایره را به ۶ مثلث تقسیم می‌کنیم در این حالت ۶ مثلث متساوی الاضلاع بدست می‌آید زیرا محیط دایره متناظر با ۳۶۰ درجه است که اگر تقسیم بر ۶ گردد زاویه داخلی برابر با ۶۰ درجه خواهد شد و لذا دو زاویه دیگر هر مثلث هم که با هم برابرند ۶۰ درجه خواهند شد بنابراین هر ضلع این ۶ ضلعی طول مساوی با شعاع یعنی ۱ خواهند داشت s=1 . سپس ارتفاع مثلث که همان x است را محاسبه نموده و پس از آن y که همان تفاضل شعاع دایره (عدد ۱) و ارتفاع است را بدست می‌آوریم. با داشتن y و s/2 می‌توانیم وتر Z را محاسبه کنیم که همان s جدید است 2n برابر طول آن هم برابر خواهد بود با محیط دایره واحد یا همان 2π لذا n ضرب در Z برابر است با تقریبی از π در گام بعدی s جدید جایگزین s قبلی می‌گردد و n دو برابر خواهد شد یعنی این‌بار دایره به ۱۲ مثلث تقسیم می‌شود با داشتن s مقدار x محاسبه می‌گردد با داشتن x مقدار y محاسبه می‌شود و با داشتن s,y مقدار z بدست می‌آید پس از آن با ضرب نمودن n در Z تقریب جدیدی از π بدست خواهد آمد. این روش که در کد زیر نشان داده شده از جمله قدیمی‌ترین روش‌ها برای محاسبه عدد π است که سرعت همگرایی مناسبی نیز دارد. با ۱۰۰ تکرار می‌توان عدد π را تا ۵۰ رقم اعشار محاسبه نمود.

```
digits(50);
clearvars
n = 6;
s = 1;

for h=1:100
    x = sqrt(1-(s/2)^2);
    y = 1-x;
    z = sqrt(((s/2)^2) + y^2);
    p = n*z;
    s = z;
    n = n * 2;
end

vpa(p)
```

مساله مانتی‌هال : یکی از مسائل جالب در مبحث احتمالات است. این مسئله اولین بار در نامه‌ای از استیو اسلوین به مجله آماردان آمریکایی در سال ۱۹۷۵ مطرح شد این مساله بر اساس یک مسابقه تلویزیونی آمریکایی به نام «بیا معامله کنیم» طرح‌ریزی شده و نامش را نیز از نام مجری این مسابقه، مانتی هال، گرفته است. مسئله مانتی هال با وجود سادگی بسیار گیج کننده است و بسیاری از افراد در برخورد اول به جواب نادرست می‌رسند و بطور جدی از آن جواب دفاع می‌کنند. این مساله، به اشکال دیگری نیز ظاهر شده که دو نمونه از «مساله زندانبان و سه زندانی» و مساله سه کارت است.

فرض کنید که در یک مسابقه تلویزیونی شرکت کرده‌اید و میان سه در باید یکی را انتخاب کنید. پشت یکی از درها یک جایزه است و پشت دو در دیگر دو پوچ است. شما یکی از درها را انتخاب می‌کنید (مثلاً در شماره ۱). مجری برنامه که از قبل می‌داند

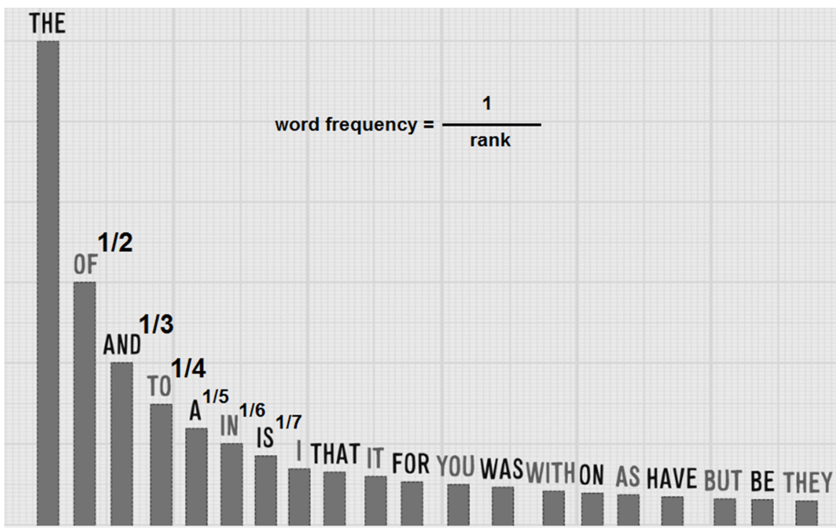
پشت هر در چه چیزی است، در دیگری را باز می‌کند (مثلاً در شماره سه) و به شما نشان می‌دهد که پشتش پوچ است. بعد از شما می‌پرسد که «می‌خواهید در شماره ۱ را با شماره ۲ عوض کنید؟» آیا به نفع شماست که انتخابتان را عوض کنید؟

در واقع پاسخ درست آنست که اگر شرکت کننده جواب خود را عوض کند شانس موفقیت او از یک سوم به دو سوم افزایش می‌یابد. برنامه زیر این روند را شبیه‌سازی می‌کند.

```
clc
clearvars
win = 0;
nwin = 0;
for cnt=1:1000
    car = randi(3);
    guess = randi(3);
    switch car
        case 1
            switch guess
                case 1
                    nguess = 3;
                case 2
                    nguess = 1;
                case 3
                    nguess = 1;
            end
        case 2
            switch guess
                case 1
                    nguess = 2;
                case 2
                    nguess = 3;
                case 3
                    nguess = 2;
            end
        case 3
            switch guess
                case 1
                    nguess = 3;
                case 2
                    nguess = 3;
                case 3
                    nguess = 2;
            end
    end

    if(guess == car)
        win = win + 1;
    end
```

قانون زیف Zipf's law : قانون زیف یک فرمول تجربی است که اگرچه اولین‌بار در زبانشناسی مورد توجه قرار گرفت اما می‌توان در بسیاری از مسائل طبیعی نظیر جمعیت شهرها، شدت طوفان‌های خورشیدی، دنباله پروتئین‌ها و گیرنده‌های ایمنی، ترافیک وب‌سایت‌ها، شدت زمین‌لرزه‌ها، تعداد دفعات ارجاع به مقالات علمی، نام‌های خانوادگی، الگوی تحریک شبکه‌های عصبی، مواد اولیه در کتاب‌های آشپزی، تعداد تماسهای تلفنی دریافتی اشخاص، قطر دهانه‌های ماه و ده‌ها مورد دیگر ردپای قانون زیف را دید. در زبانشناسی قانون زیف به این ترتیب است که اگر کلمات را بر اساس دفعات تکرار آنها در متون مرتب کنیم پر تکرارترین کلمه مشخص می‌گردد مثلا در زبان انگلیسی کلمه the. دومین کلمه پرتکرار که of می‌باشد حدود یک دوم the تکرار شده است سومین کلمه پرکاربرد and یک سوم the تکرار شده است و الی آخر در واقع تعداد تکرار یک کلمه با معکوس رتبه آن متناسب است.



این الگو تقریبا برای همه زبانها یکسان است حتی برای زبان‌های باستانی که هنوز دانشمندان موفق به ترجمه متون آن نشده‌اند. با استفاده از متلب می‌توان به تحقیق در خصوص قانون زیف پرداخت. اگر در بردار A اعداد را با توزیع زیف قرار داشته باشند کد زیر تعداد تکرار هر عدد را مشخص نموده و نسبت آن را به بشترین تکرار در بردار W قرار می‌دهد. سپس W را بصورت نمودار لگارتمی ترسیم می‌نماید اگر داده‌ها دارای توزیع زیف باشند نمودار حاصل بصورت خط خواهد بود.

```
UA = unique(A)
for k = 1:length(UA)
    UA(2,k)  = sum(A(1:length(A))==UA(1,k));
end

[W,M]=sort(UA(2,:),'descend');
W = W./W(1);

loglog(W);
```

تابع گاما : توسیع (وسیع نمودن-تعمیم دادن) تابع فاکتوریل بر اعداد حقیقی است و بصورت زیر نشان داده می‌شود:

$$\Gamma(z) = \int_0^\infty t^{z-1} e^{-t}\, \mathrm{d}t$$

$\Gamma(z) = (z-1)!$ z عدد صحیح

$$\Gamma(z+1) = z.\Gamma(z) = z.(z-1)!$$

در متلب بصورت زیر استفاده می‌شود

```
>> gamma(5)

ans =

    24
```

تبدیل لاپلاس : یک تبدیل انتگرالی خطی است که در حل معادلات دیفرانسیل خطی کاربرد دارد. این تبدیل با فرمول زیر مشخص می‌گردد.

$$F(s) = \mathcal{L}\{f(t)\} = \int_0^\infty e^{-st} f(t)\, dt.$$

در متلب تبدیل لاپلاس به شکل زیر استفاده می‌گردد.

```
>> syms x
>> f = x^2;
>> laplace(f)

ans =

2/s^3
```

تبدیل فوریه : یک تبدیل بسیار پرکاربرد است که یک سیگنال را از حوزه زمان به حوزه فرکانس (بسامد) تبدیل می‌کند. در واقع تبدیل فوریه یک سیگنال را بصورت جمع تعدادی سیگنال متناوب (سینوسی-مثلثاتی) با فرکانس‌های مختلف و ضرایب

وزنی مختلف تبدیل می‌کند. تبدیل فوریه در دو حوزه پیوسته و گسسته انجام می‌پذیرد. فرمول تبدیل فوریه پیوسته و گسسته به شکل زیر است.

$$F(\omega) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(t) e^{-i\omega t}\, dt$$

پیوسته

$$X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N} kn} \qquad k = 0, \dots, N-1$$

گسسته

در متلب می‌توان از تبدیل فوریه سریع استفاده نمود که روش سریع و موثر برای انجام تبدیل فوریه است. دستور $y = fft(X)$ برای محاسبه تبدیل فوریه سریع گسسته یا discrete Fourier transform (DFT) بکار می‌رود. توابع اصلی در جدول زیر نشان داده شده است.

fft	Fast Fourier transform
fft2	2-D fast Fourier transform
fftn	N-D fast Fourier transform
fftshift	Shift zero-frequency component to center of spectrum
fftw	Define method for determining FFT algorithm
ifft	Inverse fast Fourier transform
ifft2	2-D inverse fast Fourier transform
ifftn	Multidimensional inverse fast Fourier transform

در کد زیر ابتدا متغیر t را تعریف می‌کنیم سپس سیگنال x(t) را تعریف می‌کنیم پس از آن از سیگنال تبدیل فوریه گرفته و نشان می‌دهیم. در مثال دوم متغیرهای x و y اعلان شده و تابع نمایی f تعریف می‌شود سپس تبدیل فوریـــــه انجام می‌گیرد

```
>> syms t;
x_t=sin(t^2);
y=fourier(x_t)

y =

(2^(1/2)*pi^(1/2)*(cos(w^2/4) - sin(w^2/4)))/2
```

```
>> syms x y
f = exp(-x^2);
fourier(f, x, y)

ans =

pi^(1/2)*exp(-y^2/4)
```