



# Artificial Intelligence

مریم صمصام 9728123  
ثنا نوری 9731003  
استاد: دکتر چیترا دادخواه

# تشخیص بیماری کرونا

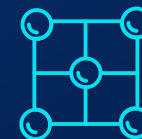
برنامه در رابطه با تشخیص بیماری کرونا در کنار بیماری های دیگر مثل سل (Tuberculosis) ، آنفولانزا (Influenza) ، سرماخوردگی (Cold) ، بیماری انسداد ریوی مزمن (COPD) و آلرژی فصلی (Seasonal Allergies) است به اینصورت که بیمار به سؤالاتی که از او پرسیده می شود پاسخ می دهد و در صورت تشخیص بیماری، نام آن چاپ می شود و در غیر اینصورت "No answers found" را چاپ میکند.

برنامه به سه قسمت تقسیم می شود:

- data.csv : در این قسمت تمام علائم بیماری های گفته شده بصورت جدولی درآمده اند.
- main.pl : در این قسمت قوانین و fact ها را از برنامه ی knowledgebase دریافت میکند و با توجه به آنها از بیمار سوال می پرسد و با توجه به پاسخ های بیمار، بیماری او را تشخیص می دهد.
- knowledgebase.pl : در این قسمت از برنامه علایم بیماری ها از جدول data.csv دریافت شده و و با ساخت درخت تصمیم ، قوانین را تعریف می شوند.

علائم بیماری ها

data



1		covid_19	influenza	cold	seasonal_allergies	tuberculosis	copd	asma
2	fever	often	often	rare	sometimes	often	often	rare
3	fatigue	sometimes	sometimes	rare	often	often	often	rare
4	cough	often	often	sometimes	often	often	often	often
5	sneezing	no	no	often	often	no	no	no
6	general_pain	sometimes	often	often	no	often	often	no
7	stuffing_nose	rare	sometimes	often	often	no	no	no
8	headache	sometimes	often	rare	sometimes	no	no	rare
9	sore_throat	sometimes	sometimes	often	no	no	no	no
10	chills	sometimes	often	rare	no	sometimes	sometimes	no
11	difficulty_breathing	sometimes	rare	rare	often	often	often	often
12	diarrhoea	rare	sometimes	no	no	no	no	no
13	loss_of_taste_smell_appetite	often	rare	sometimes	sometimes	often	no	no
14	running_nose	rare	sometimes	often	often	no	no	no
15	exhaustion	sometimes	often	no	no	no	no	sometimes
16	vomit	sometimes	sometimes	no	no	no	no	no
17	itchy_nose	no	sometimes	often	often	no	no	no
18	pink_eye	sometimes	rare	sometimes	sometimes	no	no	no
19	tiredness	often	often	sometimes	rare	usually	no	sometimes
20	sweat	no	no	no	no	often	no	no
21	fast_breath	rare	no	no	no	no	sometimes	often
22	wheezing	no	no	no	no	sometimes	often	often
23	chest_pressure	often	no	no	no	often	often	no
24	weight_loss	no	no	no	no	often	often	no
25	weakness	sometimes	often	sometimes	often	often	often	rare
26	common_in_seniors	often	sometimes	no	no	no	sometimes	often
27	common_in_children	rare	sometimes	often	sometimes	sometimes	sometimes	often
28	common_in_men	sometimes	sometimes	rare	rare	often	sometimes	sometimes
29	common_in_women	sometimes	sometimes	often	often	sometimes	sometimes	sometimes

بیماری ها

# main.pl



main.pl

```
run :-  
    write('\e[H\e[2J'),  
    write("Welcome!"),nl,  
    repeat,  
    write("> Enter load, consult or quit"),nl,  
    write("> "),  
    read(Command),  
    do(Command),  
    Command = quit.  
  
do(load) :-  
    reconsult('c:/users/lenovo/desktop/AI-1.0/knowledgebase.pl'),  
    load_rules,  
    write("> Successfully loaded"),nl,!.  
do(consult) :- solve,!.  
do(quit) :- write("Goodbye"),nl,!.  
do(X) :-  
    write(X),  
    write(" is not a legal command, try again."),nl.  
  
solve :-  
    retractall(attr_value(_,_)),  
    attributes(As),  
    find_disease(As,1),  
    disease(Disease),  
    write("Disease is "),write(Disease),nl,!.  
solve :- write("No answers found"),nl.  
  
find_disease(Attrs,Index) :-  
    length(Attrs,X),  
    L1 is Index - 1,  
    X = L1,!.  
find_disease(Attrs,Index) :-  
    not(disease(_)),  
    get(Aattrs,Index,A),  
    not(attr_value(A,_)),  
    values(A,Vs),  
    menuask(A,Vs),
```

با شروع برنامه و اجرای دستور run از کاربر خواسته می شود دستور مورد نظر خود را وارد کند

با وارد شدن دستور load، تابع load\_rules در knowledgebase فراخوانی می شود تا قوانین از روی جدول ساخته و اضافه شوند

با وارد شدن دستور quite عبارت goodbye چاپ شده و برنامه به پایان میرسد

با وارد شدن دستور consult، slove فراخوانی می شود و سپس find\_disease که طی آن به ترتیب علائم مختلف از کاربر پرسیده شده و طبق قوانین بیماری تشخیص داده می شود.





```

find_disease(Attrs,Index) :-
    length(Attrs,X),
    L1 is Index - 1,
    X = L1,!.

find_disease(Attrs,Index) :-
    not(disease(_)),
    get(Attrs,Index,A),
    not(attr_value(A,_)),
    values(A,Vs),
    menuask(A,Vs),
    Index1 is Index + 1,
    find_disease(Attrs,Index1),!.

find_disease(Attrs,Index) :-
    not(disease(_)),
    get(Attrs,Index,A),
    attr_value(A,_),
    Index1 is Index + 1,
    find_disease(Attrs,Index1).

find_disease(_,_) .

menuask(Attr,MenuList) :-
    write('What is the value for '),
    write(Attr),write('?'),nl,
    write(MenuList),nl,
    read(V),
    check_val(V,Attr,MenuList),
    asserta(attr_value(Attr,V)).

check_val(X,_,MenuList) :- member(X, MenuList), !.
check_val(X,Attr,MenuList) :-
    write(X),write(' is not a legal value, try again. '),nl,
    menuask(Attr,MenuList).

```

در این قسمت با توجه به درخت ساخته شده در knowledgebase و قوانین، از کاربر مقدار علائم مورد نیاز(attr) را سوال میپرسد و سعی در پیدا کردن بیماری می کند.

با توجه به قوانین مقدار attr خواسته شده را از کاربر میپرسد و از او میخواهد مقدار آنرا از آرایه ی گفته شده (MenuList) انتخاب کند.

مقداری که از کاربر برای Attr دریافت کرده (x) را چک میکند که آیا برابر یکی از درایه های آرایه (MenuList) مورد نظر است یا خیر. که اگر نبود از او میخواهد که دوباره پاسخ دهد.

# knowledgebase.pl (پایگاه دانش)



knowledgebase.pl

```
%predicates
%attribute( fever ).
%attr_value_disease( fever, often, covid_19 ).
%attr_value( fever, sometimes ).
%disease( test ).
```

```
prepare :-
    dynamic
        attribute/1, attr_value/2,
        attr_value_disease/3, disease/1.
```

برای اینکه بتوان با استفاده از predicate ها در زمان اجرا  
ثانون اضافه کرد لازم است به صورت dynamic تعریف  
شوند

```
load_rules :-
    prepare,
    %remove previous data
    retractall( attribute( _ ) ),
    retractall( attr_value_disease( _ , _ , _ ) ),
    retractall( disease( _ ) ),
    retractall( attr_value( _ , _ ) ),
    %read data from file
    csv_read_file( 'c:/users/lenovo/desktop/AI-1.0/AI.final (3).csv', Rows ),
    maplist( rows_to_list, Rows, Table ),
    get( Table, 1, Diseases ),
    add_attrs( Table, Diseases, 2 ),
    attributes( A ),
    write( A ), nl,
    make_tree( 1, [ ] ).
```

داده های قبلی از پایگاه حذف می شوند تا داده های  
جدید قرار بگیرند

```
rows_to_list( Row, List ) :- Row =.. [ row | List ].
```

Attribute ها از روی جدول خوانده شده و  
اضافه می شوند

```
%get element of Array at Index
get( Array, 1, Value ) :- Array = [ X | _ ], Value = X, !.
get( Array, Index, Value ) :-
    Index1 is Index - 1,
    Array = [ _ | X ],
    get( X, Index1, Value ).
```

درخت ساخته می شود

```
%add all attr_value_disease predicates from table
add_attrs( Table, _, Index ) :-
```

با استفاده از این تابع داده های داخل  
جدول خوانده می شوند

# knowledgebase.pl (پایگاه دانش)



knowledgebase.pl

```
%add all attr_value_disease predicates from table
add_attrs(Table,_,Index) :-
    length(Table,X),
    X = Index,!.
add_attrs(Table,Diseases,Index) :-
    get(Table,Index,Row),
    Row = [Attr|Values],
    assertz(attribute(Attr)),
    add_attr_value(Diseases,Attr,Values,1),
    Index1 is Index + 1,
    add_attrs(Table,Diseases,Index1).

%add all attr_value_disease predicates from row
add_attr_value(_,_,Values,Index) :-
    length(Values,X),
    L1 is Index - 1,
    X = L1,!.
add_attr_value(Diseases,Attr,Values,Index) :-
    get(Values,Index,V),
    L1 is Index + 1,
    get(Diseases,L1,D),
    assertz(attr_value_disease(Attr,V,D)),
    add_attr_value(Diseases,Attr,Values,L1).

%loop on all attributes to make tree
make_tree(Attr_index,Path_in) :-
    attributes(As),
    get(As,Attr_index,A),
    %write(A),nl,
    values(A,Vs),
    %write(Vs),nl,
    make_branch(A,Vs,Attr_index,Path_in,1).

%find all attributes added as a fact
attributes(Attrs) :- findall(A,attribute(A),Attrs).

%find all values for Attr
values(Attr,Values) :-
    findall(V,attr_value_disease(Attr,V,_),Vs),
```

با توجه به نیاز به اجرای فعالیت های تکراری روی عناصر ارایه های و علائم مختلف، اکثر predicate ها به صورت بازگشتی نوشته شده اند.

در add\_attrs ارایه سطر های خوانده شده از جدول در متغیر Table به ترتیب طی شده و علائم از آن ها خوانده می شود و هر یک به صورت attribute(attr) اضافه می شوند تا بعدا از آن ها استفاده شود.

همچنین در add\_attr\_value هر یک از خانه های جدول (بجز خود بیماری ها و علائم) به صورتی سه تایی (attribute,value,disease) اضافه می شوند

در make\_tree به ترتیب attribute ها انتخاب شده و هر یک از value های آن ها برای ساخت درخت در نظر گرفته می شود.

برای هر attribute که با اندیشش در آرایه attribute ها شناخته می شود، یک مسیری برای ورود به آن وجود دارد که طبق این مسیر و مقادیر انتخاب شده در هر مرحله، مسیر برای مرحله بعدی و ساخت سطح بعدی درخت مشخص می شود.



# knowledgebase (پایگاه دانش)



knowledgebase.pl [modified]

```
possible_diseases(Path, Index, D_in, D_out) :-
    length(Path, X),
    L1 is Index - 1,
    X = L1,
    (X == 0 ->
        findall(D, attr_value_disease(_, _, D), D)
    );
    D = D_in,
    sort(D, D_out), !.

possible_diseases(Path, Index, D_in, D_out) :-
    get(Path, Index, Att_val),
    Att_val = [Attr, Value],
    findall(D, attr_value_disease(Attr, Value, D), D1),
    length(D_in, L),
    (L == 0 -> D2 = D1;
        intersection(D1, D_in, D2)),
    Index1 is Index + 1,
    possible_diseases(Path, Index1, D2, D_out).

add_rule(Path) :-
    length(Path, L),
    get(Path, L, Disease),
    assertz((disease(Disease)) :- call_all(Path, 1)).

call_all(Path, Index) :-
    length(Path, X),
    L1 is Index,
    X = L1, !.

call_all(Path, Index) :-
    get(Path, Index, Att_val),
    Att_val = [Attr, Value],
    attr_value(Attr, Value),
    Index1 is Index + 1,
    call_all(Path, Index1).
```

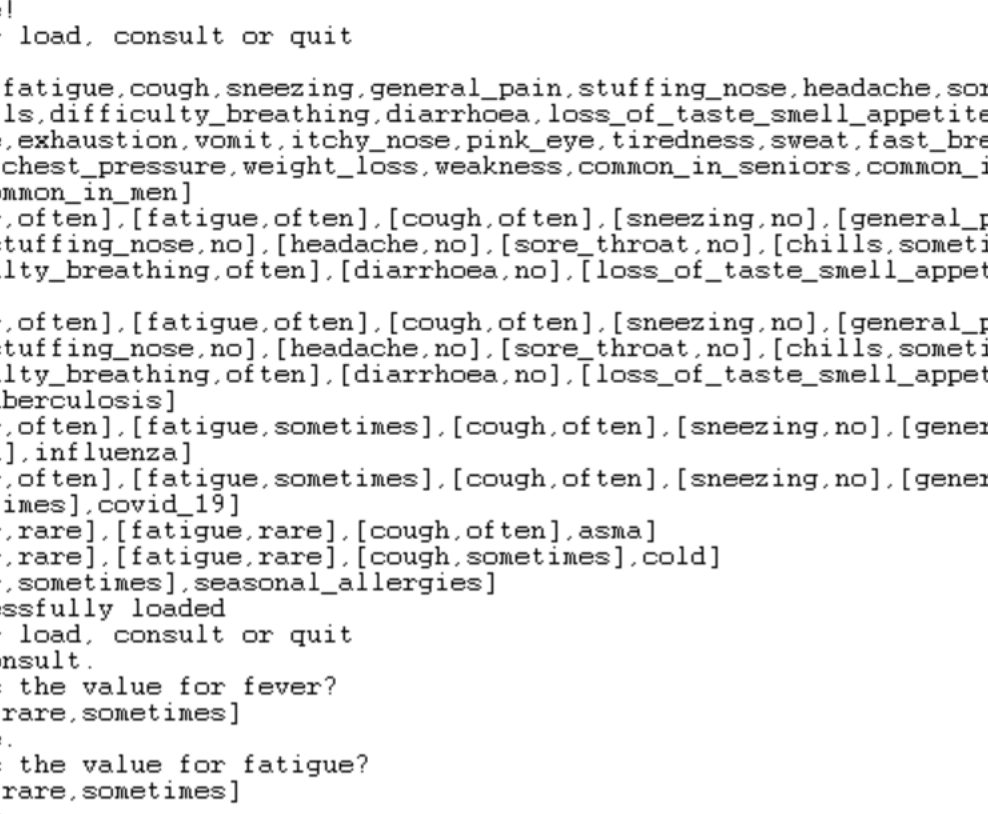
بعد از تکمیل هر مسیر از ریشه و رسیدن به یک بیماری، این مسیر به predicate add\_rule داده می شود تا قانون مربوط به آن به پایگاه اضافه شود به اینصورت که بیماری، بیماری انتهای مسیر است اگر تمام attribute ها را با مقادیر آن مسیر داشته باشد باشد.

## توضیحات

- سایر predicate ها به منظور اجرای درست برنامه و برحسب نیاز تعریف شده اند.
- برای نوشتن این برنامه از زبان prolog و محیط برنامه نویسی SWI-prolog با ورژن 8.2.4 استفاده شده است.
- برای اجرا بعد از باز کردن نرم افزار swi لازم است دو فایل knowledge.pl و main.pl به ترتیب consult شوند. (در تب file گزینه consult)
- برای اجرای درست برنامه لازم است آدرس فایل knowledge.pl که در فایل main.pl استفاده شده، و آدرس فایل data که در فایل knowledge.pl استفاده شده برحسب موقعیت آن ها تغییر کند.



# خروجی



```
SWI-Prolog (AMD64, Multi-threaded, version 8.2.4)
File Edit Settings Run Debug Help
Welcome!
> Enter load, consult or quit
> load.
[fever,fatigue,cough,sneezing,general_pain,stuffing_nose,headache,sore_throat,chills,difficulty_breathing,diarrhoea,loss_of_taste_smell_appetite,running_nose,exhaustion,vomit,itchy_nose,pink_eye,tiredness,sweat,fast_breath,whooping_cough,chest_pressure,weight_loss,weakness,common_in_seniors,common_in_children,common_in_men]
[[fever,often],[fatigue,often],[cough,often],[sneezing,no],[general_pain,often],[stuffing_nose,no],[headache,no],[sore_throat,no],[chills,sometimes],[difficulty_breathing,often],[diarrhoea,no],[loss_of_taste_smell_appetite,no],[cough,no],[covid_19]]
[[fever,often],[fatigue,often],[cough,often],[sneezing,no],[general_pain,often],[stuffing_nose,no],[headache,no],[sore_throat,no],[chills,sometimes],[difficulty_breathing,often],[diarrhoea,no],[loss_of_taste_smell_appetite,often],[tuberculosis]]
[[fever,often],[fatigue,sometimes],[cough,often],[sneezing,no],[general_pain,often],[influenza]]
[[fever,often],[fatigue,sometimes],[cough,often],[sneezing,no],[general_pain,sometimes],[covid_19]]
[[fever,rare],[fatigue,rare],[cough,often],[asma]]
[[fever,rare],[fatigue,rare],[cough,sometimes],[cold]]
[[fever,sometimes],[seasonal_allergies]]
> Successfully loaded
> Enter load, consult or quit
> |: consult.
What is the value for fever?
[often,rare,sometimes]
|: rare.
What is the value for fatigue?
[often,rare,sometimes]
|: rare.
What is the value for cough?
[often,sometimes]
|: often.
Disease is asma
```

با اجرای دستور run برنامه شروع شده و سپس با `load` درخت ساخته و نمایش داده می شود. در خط اول خروجی، در یک آرایه تمام `attribute` ها (علائم) خوانده شده از جدول چاپ می شود و در ادامه مسیرهای از درخت که به نتیجه (تشخیص بیماری) می رسند چاپ شده است. هر مسیر به صورت آرایه ای از `[attr,value]` ها به ترتیب از ریشه به پایین و در انتها بیماری تشخیص داده شده است.

سپس با زدن **consult** سوالات از کاربر پرسیده می شود و با استفاده از قوانین استخراج شده از جدول بیماری مشخص می شود



```
SWI-Prolog (AMD64, Multi-threaded, version 8.2.4)
File Edit Settings Run Debug Help
[[fever,sometimes],seasonal_allergies]
> Successfully loaded
> Enter load, consult or quit
> |: consult.
What is the value for fever?
[often,rare,sometimes]
|: rare.
What is the value for fatigue?
[often,rare,sometimes]
|: rare.
What is the value for cough?
[often,sometimes]
|: often.
Disease is asma
> Enter load, consult or quit
> |: consult.
What is the value for fever?
[often,rare,sometimes]
|: often.
What is the value for fatigue?
[often,rare,sometimes]
|: sometimes.
What is the value for cough?
[often,sometimes]
|: often.
What is the value for sneezing?
[no,often]
|: no
|: .
What is the value for general_pain?
[no,often,sometimes]
|: sometimes.
Disease is covid_19
> Enter load, consult or quit
> |: quit.
Goodbye
true .
```

بیماری دوم هم به همین ترتیب با زدن دوباره دستور `consult` تشخیص داده می شود.  
در نهایت با زدن دستور `quit` برنامه به اتمام می رسد.

# THANKS



[maryamsamsam@email.kntu.ac.ir](mailto:maryamsamsam@email.kntu.ac.ir)  
[sananouri@email.kntu.ac.ir](mailto:sananouri@email.kntu.ac.ir)