****

**Nanyang Technological University**

**School of Computer Science and Engineering**

**CZ3005 Artificial Intelligence LAB 3 REPORT**

**Name:**

LEE SUYEON

**Matric ID:**

N1801612A

**Group Number:**

TS2

**Submission Date:**

15, April, 2019

**0. Assignment**

<Assignment 2: Kid’s day at school>

Assume that the prolog script is a parent, trying to know about a kid’s day at school. The prolog script should converse intelligently with the kid as follows. The prolog script should ask a question to the kid that kid can answer in yes or no. Depending on whether the answer is yes or no, prolog script should ask a related question or another random question.

**1. Code Implementation**

: Omitted comments in this section. There are detailed comments in code file(softcopy submitted online), and detailed explanation for code is in section 2 of the report.

**:- dynamic did/1.**

**:- dynamic didnot/1.**

**ask(0):-**

write("Welcome Home! Was it a good day at school? (y./n.:) "),read(KidAns),(KidAns==y -> validate\_and\_query\_options([play]);KidAns==n -> validate\_and\_query\_options([test])).

**ask(Y):-**

generate\_options(Y,L),validate\_and\_query\_options(L).

**generate\_options(Y,L):-**

did(Y),((play(PlayList),member(Y,PlayList),findall(X,play\_related(X,Y),L1),random\_permutation(L1,L),write("Okay, did you play "));

(eat(EatList),member(Y,EatList),findall(X,eat\_related(X,Y),L1),random\_permutation(L1,L),write("Okay, did you eat "));

(test(TestList),member(Y,TestList),findall(X,test\_related(X,Y),L1),random\_permutation(L1,L),write("Okay, did you test "));

(make(MakeList),member(Y,MakeList),findall(X,make\_related(X,Y),L1),random\_permutation(L1,L),write("Okay, did you make "));

(watch(WatchList),member(Y,WatchList),findall(X,watch\_related(X,Y),L1),random\_permutation(L1,L),write("Okay, did you watch "));

(exercise(ExerciseList),member(Y,ExerciseList),findall(X,exercise\_related(X,Y),L1),random\_permutation(L1,L),write("Okay, did you do ")); (like(LikeList),member(Y,LikeList),findall(X,like\_related(X,Y),L1),random\_permutation(L1,L),write("You must had a great day! Did you "));

(dislike(DislikeList),member(Y,DislikeList),findall(X,dislike\_related(X,Y),L1),random\_permutation(L1,L),write("I'm sorry to hear that... Did you ")));

didnot(Y),findall(X,total\_related(X,Y),L1),[H|T]=L1,write("Then... Did you "),((like(LikeList),member(H,LikeList),findall(X,like\_related(X,H),L));

(dislike(DislikeList),member(H,DislikeList),findall(X,dislike\_related(X,H),L))).

**validate\_and\_query\_options(L):-**

findall(X,did(X),DidList),findall(X,didnot(X),DidnotList),append(DidList,DidnotList,History),list\_to\_set(L,S),list\_to\_set(History,H),subtract(S,H,Valid),(length(Valid,0) -> abort;random\_member(X,Valid)),(

X==play -> (write("Awesome! Did you "),print(X));

X==test -> (write("What's the matter? Did you "),print(X));

print(X)),write("? (y./n./q.:) "), read(KidAns),(KidAns==q -> abort;KidAns==y -> assert(did(X));

KidAns==n -> assert(didnot(X))), ask(X).

**total\_related(X,Y):-**

like(L),member(X,L),member(Y,L);

dislike(L),member(X,L),member(Y,L);

play(L),member(X,L),member(Y,L);

eat(L),member(X,L),member(Y,L);

test(L),member(X,L),member(Y,L);

make(L),member(X,L),member(Y,L);

watch(L),member(X,L),member(Y,L);

exercise(L),member(X,L),member(Y,L).

**like\_related(X,Y):-**

like(L),member(X,L),member(Y,L).

**dislike\_related(X,Y):-**

dislike(L),member(X,L),member(Y,L).

**play\_related(X,Y):-**

play(L),member(X,L),member(Y,L).

**eat\_related(X,Y):-**

eat(L),member(X,L),member(Y,L).

**test\_related(X,Y):-**

test(L),member(X,L),member(Y,L).

**make\_related(X,Y):-**

make(L),member(X,L),member(Y,L).

**watch\_related(X,Y):-**

watch(L),member(X,L),member(Y,L).

**exercise\_related(X,Y):-**

exercise(L),member(X,L),member(Y,L).

**like**([play,eat,watch,swim,take\_nap,talk\_friends,read\_book,play\_game,take\_rest]).

**dislike**([test,exercise,make,study,clean\_up,do\_assignment,check\_test\_score,draw\_picture,memorize\_words]).

**play**([play,sandbox,toys,trains,dolls,half\_hour,one\_hour,with\_friends,with\_teacher]).

**eat**([eat,cake,candy,sandwich,pizza,fries,with\_spoon,with\_fork,after\_washing\_hands,and\_clean\_up]).

**test**([test,math,science,alphabets,open\_book,online,easy\_level,hard\_level]).

**make**([make,bread,doll,cookie,with\_friend,for\_dad,for\_mom,for\_sister]).

**watch**([watch,movie,animation,musical,boring\_one,interesting\_one,scary\_one,two\_hours,three\_hours]).

**exercise**([exercise,running,soccer,basketball,well,win,hurt]).

**did(nothing).**

**didnot(nothing).**

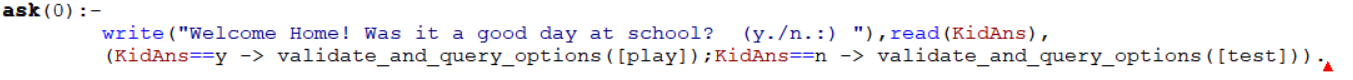
**a.**

**2. Code Explanation**

1. Declare dynamic variables. "did" and "didnot" has different value each time, and it stores old values.



2. You can start program by "ask(0)." query. Kid will be asked the mood of the day. In this query, kid can only answer y./n. (not q.). Asking this because following questions will be different depend on kid’s mood.



1) If kid answers y, it means kid did activities that he/she likes at school. Those activities are clarified as "like([])" fact below. You will first ask if kid played at school.

2) If kid answers n, it means kid did activities that he/she does not like at school. Those activities are clarified as "dislike([])" fact below. You will first ask if kid tested at school.



3. You can keep asking according to conditions clarified in generate\_options and validate\_and\_query\_options.



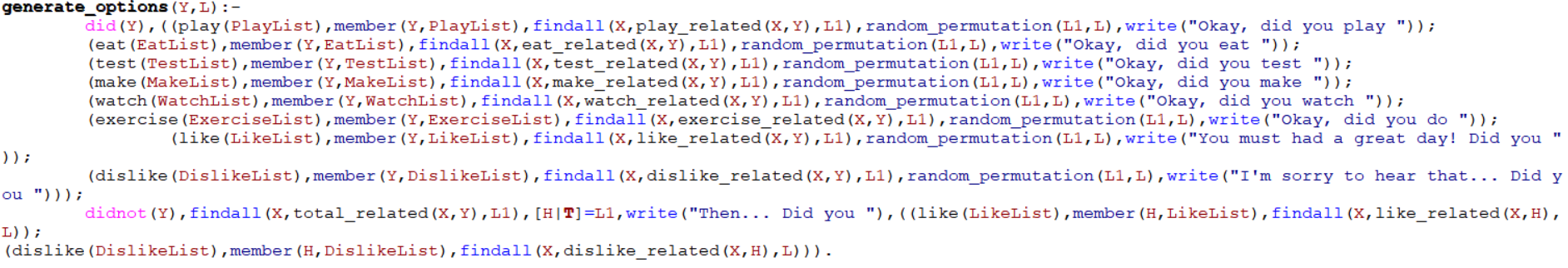
4. Parameter Y is passed from validate\_and\_query\_options. It means kid was asked if he/she did activity Y.

1) If kid answered he/she did Y, then check facts to find next question. If kid did play/ eat/ test/ make/ watch/ exercise, you can ask relevant question by returning list of each facts. There are relevant behaviors in each facts. (For example, for “play” fact, look picture below.) If kid did other activities, print proper response depending on kid's mood and ask random activity question.



2) If kid answered he/she not did Y, ask other random activities depend on kid's mood. For example, if kid's mood is good, you only allowed to ask activities which kid likes. If kid's mood is bad, you only allowed to ask activities which kid does not like.

To find out activities related to kid's mood, needed to look up "like" and "dislike" facts. Each activity fact stores main behavior as first element. For example, play([play, train, ...]). (look picture above.) So we first get activity's name by using total\_related(X,Y). Then get head from the activity list, and search if the activities is in "like" fact or in "dislike" fact.



5. To avoid asking same questions, created list named "Valid" by subtracting Input list from History. Able to make History list by union DidList, which is all(old and new) values of did(X), and DidnotList, which is all values of didnot(X). If Valid is empty list, it means you can't ask questions no more(used all facts, etc.). Then stop excution using abort. If Valid is not empty, then get a random element X from Valid list.



1) If X is 'play', that means it is right after mood question and kid mood was good. Print proper response and ask if kid played.



2) If X is 'test', that means it is right after mood question and kid mood was not good. Print proper response and ask if kid tested.



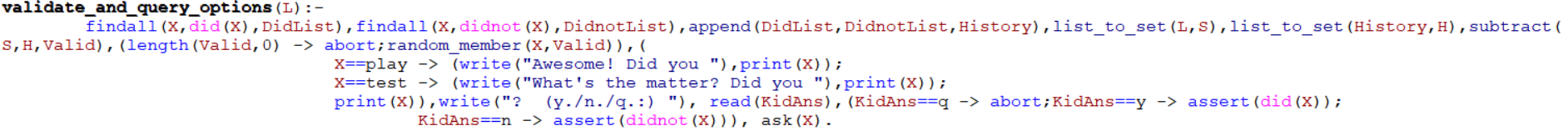
3) Else, that means it is not a first question. In this case, already printed proper response at generated\_options above. So just ask what kid did by using print(X).

Kid can answer y./n./q. from now on.

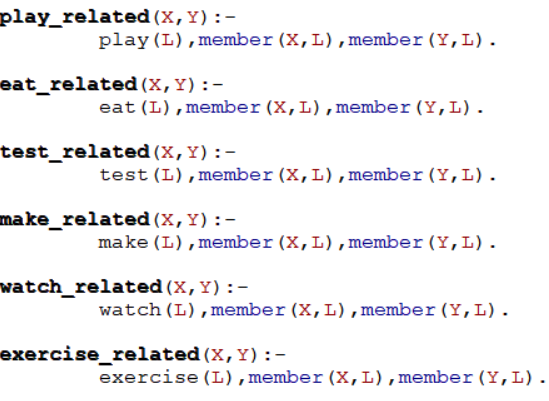
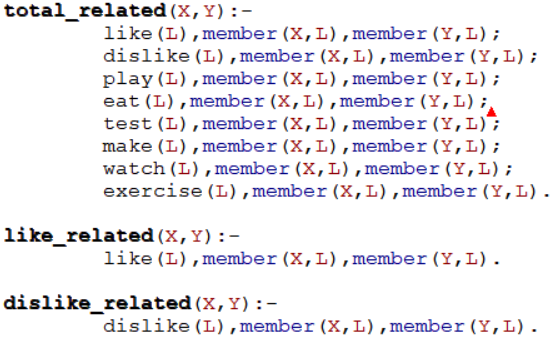
1) If kid answers y, it means kid did the activity. Store X as did(X) using assert().

2) If kid answers n, it means kid not did the activity. Store X as didnot(X) using assert().

3) If kid answers q, stop execution using abort.



6. Clarify related facts to find all X element which is related with Y.



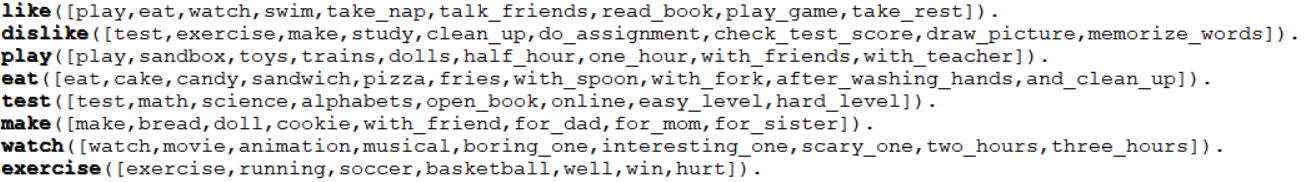
7. Below picture is declared facts.

- "like" has activities that kid likes. If he/she did those things, kid's mood is good.

- "dislike" has activities that kid does not like. If he/she did those things, kid's mood is not good.

- Each "play","eat","test","make","watch","exercise" activities has its own facts which will be used to ask relevant questions.

- If activities which does not have own facts asked, for example asked “swim”, it will lead to ask next question as random one.

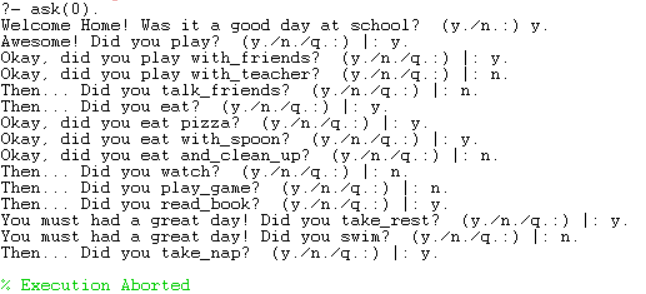


8. Initialize dynamic variables.



**3. Code Execution**

**case 1) Kid’s mood is good.**



Kid’s mood is good since answered y. for first question. So you will be able to ask activities in “like” facts.

First asked if kid played. Since kid said yes, keep asking relevant questions.(play with whom, etc.)

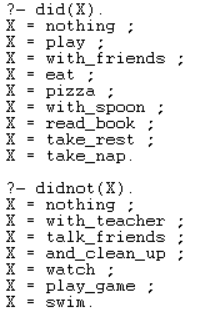
Kid answered no for playing with teacher, move to next random question until kid say yes.

When asked if kid ate, kid said yes. Similarly keep asking relevant questions.

Kid answered no for cleaning up after eating, move to next random question until kid say yes.

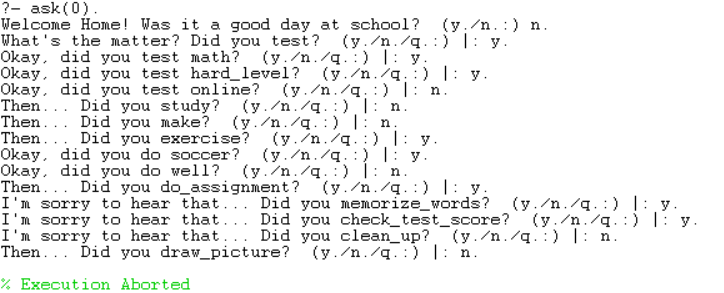
For reading book and taking rest activities, set to print proper response and move right on to the next random question. (look result picture above.)

It terminated automatically since Valid list is empty. In this case, parent asked all activities in “like” facts.



We can see what kid did and not did activities are stored properly in dynamic variables as above picture.

**case 2) Kid’s mood is not good.**



Kid’s mood is not good since answered n. for first question. So you will be able to ask activities in “dislike” facts.

First asked if kid tested. Since kid said yes, keep asking relevant questions.(subject, test level, etc.)

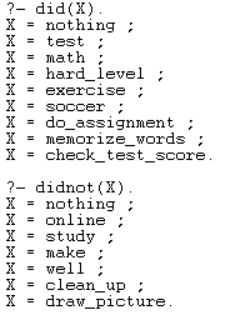
Kid answered no for online test, move to next random question until kid say yes.

When asked if kid exercised, kid said yes. Similarly keep asking relevant questions.

Kid answered no for whether playing soccer well, move to next random question until kid say yes.

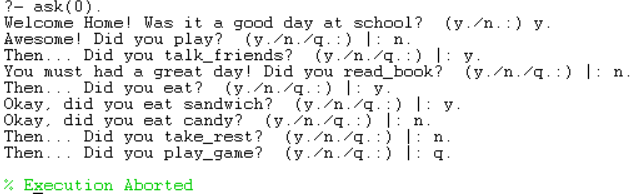
For doing assignments, memorizing words, and checking test score activities, set to print proper response and move right on to the next random question. (look result picture above.)

It terminated automatically since Valid list is empty. In this case, parent asked all activities in “dislike” facts.

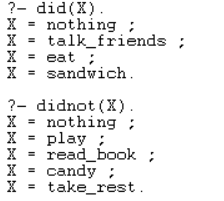


We can see what kid did and not did activities are stored properly in dynamic variables as picture above.

**case 3) Teminating by answering q.**

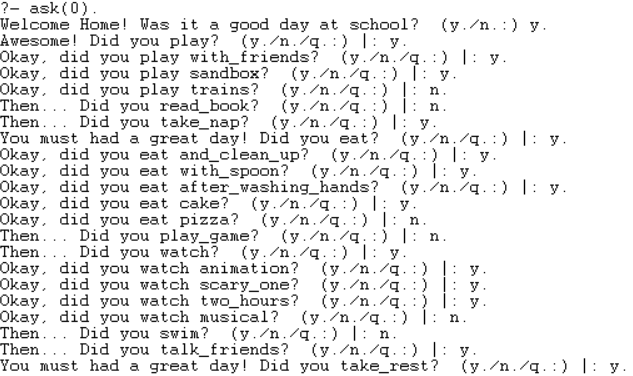


We can see that the program was terminated when kid answered q.

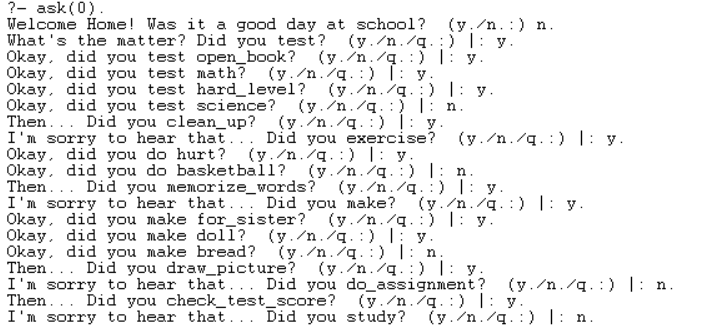


What kid did and not did activities are also stored properly in dynamic variables as picture above.

**case 1-1)** Some other examples of relevant questions for each activities when kid’s mood is good.



**case 2-1)** Some other examples of relevant questions for each activities when kid’s mood is bad.



**(End)**