```
% ======1.导入需要的内容======

clc;%清除命令窗口
clear all;%清除变量
impdata=importdata('a009.log');

event_type=impdata.textdata(:,3); % 读取event type
event_type=event_type(4:1616)%将event_type列中有空集的几行剔除

code=impdata.textdata(:,4); %读取code列
code=code(4:1616); %将code列中有空集的几行剔除


time=impdata.textdata(:,5);  %读取的time一列的每一个时刻节点;
time=time(4:1616); %剔除Time列中的空集行
time=str2num(char(time));

% ======2.先找出每个trial的位置，并存储对应位置的量======
num = 0; % num为trial序号
t0 = 976921/10000; % 第一个trial的初始时间
for i = 1:length(code) % 在code列内寻找trial('greenbox1' or 'redbox1')
    if strcmp(code{i},'greenbox1') || strcmp(code{i},'redbox1') % 寻找‘greenbox1’和↙
‘redbox1’
        num = num+1;
        trial(num).num = num; % 创建结构体，存储tria.num
        trial(num).mode = code{i}; % 存储trial.mode
        trial(num).start_time = time(i)/10000 - t0; % 存储trial.start_time
        trial_position(num) = i; % 记录每个trial的位置
   end
end

% ======3.在所有trial中寻找ISI和ITI的位置并存储其time到对应的trial======
for j = 1 : length(trial_position)
    if j < length(trial_position)
        for k = trial_position(j)+1 : trial_position(j+1)-1 % 在前（length↙
 (trial_position) -1）个trial中寻找ISI和ITI的位置
            switch code{k}
                case 'crossISI_4_8'
                    trial(j).ISI = time(k)/10000 - t0; % 存储trial.ISI
                case 'crossITI_8'
                    trial(j).ITI = time(k)/10000 - t0; % 存储trial.ITI
            end
        end
    else
        for k = trial_position(end)+1 : length(code) % 在最后一个trial中寻找ISI和ITI的位置
            switch code{k}
                case 'crossISI_4_8'
                    trial(j).ISI = time(k)/10000 - t0; % 存储trial.ISI
                case 'crossITI_8'
                    trial(j).ITI = time(k)/10000 - t0; % 存储trial.ITI
            end
        end
    end
end

% ======4.在所有trial中寻找计算并存储rating_time和rating_reponse_time======
rating_num = 0; % 记录rating_time的次序
```

```
resptime_num = 0; % 记录rating_response_time的次序
for j = 1 : length(trial_position)

    if j < length(trial_position) % 前(length(trial_position)-1)个trial
        for k = trial_position(j)+1 : trial_position(j+1)-1 % 寻找第j个trial的rating_time↙
('Display Scale')的位置并存储到trial(j)结构体
            if strcmp(code{k},'Display Scale')
                rating_num = rating_num+1;
                trial(j).rating_time{rating_num} = time(k)/10000 - t0; % 存储rating_time到↙
trial(j)结构体
                rating_position(rating_num) = k; % 记录rating_time的位置
            end
        end
        rating_num = 0; % 每一个trial寻找完成后要归零
        for m = 1:length(rating_position) % 在第j个trial的每个rating_time('Display Scale')后↙
寻找rating_response_time(距离最近的'Response')并存储到对应的trial(j)结构体
            if m < length(rating_position) % 前(length(rating_position)-1)个rating_time后寻↙
找rating_response_time
                for n = rating_position(m)+1 : rating_position(m+1)-1
                    if strcmp(event_type{n},'Response')
                        resptime_num = resptime_num+1;
                        trial(j).rating_response_time{resptime_num} = time(n)/10000 - t0↙
- trial(j).rating_time{resptime_num};
                        break % 只找最近的'Response'，寻找到就立即终止for循环
                    end
                end

            else % 在最后一个rating_time后寻找rating_response_time
                for n = rating_position(end)+1 : trial_position(j+1)-1
                    if strcmp(event_type{n},'Response')
                        resptime_num = resptime_num+1;
                        trial(j).rating_response_time{resptime_num} = time(n)/10000 - t0↙
- trial(j).rating_time{resptime_num};
                        break % 只找最近的'Response'，寻找到就立即终止for循环
                    end
                end
            end
        end
        resptime_num = 0;
        rating_position = []; % 一定要清空，不然前面的rating_position信息可能会影响后面的
                                    % 前(length(trial_position)-1)个trial


    else % 最后一个trial
        for k = trial_position(end)+1 : length(code)
            if strcmp(code{k},'Display Scale')
                rating_num = rating_num+1;
                trial(j).rating_time{rating_num} = time(k)/10000 - t0;
                rating_position(rating_num) = k;
            end
        end
        for m = 1:length(rating_position)
            if m < length(rating_position)
                for n = rating_position(m)+1 : rating_position(m+1)-1
                    if strcmp(event_type{n},'Response')
                        resptime_num = resptime_num+1;
                        trial(j).rating_response_time{resptime_num} = time(n)/10000 - t0↙
- trial(j).rating_time{resptime_num};
```

```
                        break
                    end
                end

            else
                for n = rating_position(end)+1 : length(event_type)
                    if strcmp(event_type{n},'Response')
                        resptime_num = resptime_num+1;
                        rating_response_time(resptime_num) = time(n)/10000 - t0 - trial↙
(j).rating_time{resptime_num};
                        break
                    end
                end
            end
        end
    end % 最后一个trial

end



% ======5.在所有的trial中寻找pain、vibration、difficulty和difference的值======
for j = 1 : length(trial_position)
    trial(j).pain = []; % 预分配内存，控制顺序
    trial(j).vibration = [];
    trial(j).difficulty = [];
    trial(j).difference = [];
    if j < length(trial_position) % ------------------前(length(trial_position)-1)个↙
trial-----------------
        for k = trial_position(j)+1 : trial_position(j+1)-1
            if strmatch('This pain',code{k}(8:end))
                trial(j).pain = str2num(code{k}(end-1:end));
            elseif  strmatch('This vibration',code{k}(8:end))
                trial(j).vibration = str2num(code{k}(end-1:end));
            elseif strmatch('Difficulty',code{k}(8:end))
                trial(j).difficulty = str2num(code{k}(end-1:end));
            elseif strmatch('Difference',code{k}(8:end))
                trial(j).difference = str2num(code{k}(end-1:end));
            end
        end


    else % ---------------最后一个trial----------------
        for k = trial_position(end)+1 : length(code)
            if strmatch('This pain',code{k}(8:end))
                trial(j).pain = str2num(code{k}(end-1:end));
            elseif  strmatch('This vibration',code{k}(8:end))
                trial(j).vibration = str2num(code{k}(end-1:end));
            elseif strmatch('Difficulty',code{k}(8:end))
                trial(j).difficulty = str2num(code{k}(end-1:end));
            elseif strmatch('Difference',code{k}(8:end))
                trial(j).difference = str2num(code{k}(end-1:end));
            end
        end
    end
end
```

```
% ======6.对trial结构体中的成员进行简单调整======

for i = 1:length(trial) % 分别将实验模态的'redbox1'和'greenbox1'改成'memory encoding'和'non-↙
memory encoding'
    switch trial(i).mode
        case 'redbox1'
            trial(i).mode = 'memory encoding';
        case 'greenbox1'
            trial(i).mode = 'non-memory encoding';
    end
end

for n = 1:length(trial) % 将rating_time和rating_response_time细胞数组分开保存
    trial(n).rating_time1 = []; % 预先分配内存
    trial(n).rating_reponse_time1 = [];
    trial(n).rating_time2 = [];
    trial(n).rating_reponse_time2 = [];
    trial(n).rating_time3 = [];
    trial(n).rating_reponse_time3 = [];
    switch length(trial(n).rating_time)
        case 3
            trial(n).rating_time1 = trial(n).rating_time{1,1};
            trial(n).rating_time2 = trial(n).rating_time{1,2};
            trial(n).rating_time3 = trial(n).rating_time{1,3};
        case 2
            trial(n).rating_time1 = trial(n).rating_time{1,1};
            trial(n).rating_time2 = trial(n).rating_time{1,2};
        case 1
            trial(n).rating_time1 = trial(n).rating_time{1,1};
        case 0
    end
    switch length(trial(n).rating_response_time)
        case 3
            trial(n).rating_reponse_time1 = trial(n).rating_response_time{1,1};
            trial(n).rating_reponse_time2 = trial(n).rating_response_time{1,2};
            trial(n).rating_reponse_time3 = trial(n).rating_response_time{1,3};
        case 2
            trial(n).rating_reponse_time1 = trial(n).rating_response_time{1,1};
            trial(n).rating_reponse_time2 = trial(n).rating_response_time{1,2};
        case 1
            trial(n).rating_reponse_time1 = trial(n).rating_response_time{1,1};
        case 0
    end
end

trial = rmfield(trial,{'rating_time','rating_response_time'}); % 删除trial结构体中↙
rating_time和rating_response_time两个成员变量



% ======7.将trial的分类结果写入Excel文件======
values = struct2cell(trial(:)); % 结构转细胞
headers = fieldnames(trial); % 获得标题
xlsdata = cat(2, headers, values)'; % 连接
xlswrite('trials.xlsx', xlsdata); % 写入excel文件
```