```
% *** Aidan Chin, 9/22/23, etc.

    %   *** Label carts from left to right as 1, 2, 3
    %   *** Label "collisions" A, B, C, etc. until everything is final
    %   *** do not use loops or functions
    %   *** use arrays to keep track of everything
    %   *** Formulas from M1, http://hyperphysics.phy-astr.gsu.edu/hbase/elacol2.html

clear

% ----- Getting Started -----

% givens

m = [ 300 60 240 ] % mass of cars in g from left to right
v0 = [ 36 9 -45 ] % velocity of cars in cm/s from left to right

% set up total masses for two types of collisions

m23 = m(2)+m(3); % total mass of carts 2 and 3 in g
m12 = m(1)+m(2); % total mass of carts 1 and 2 in g

% set up the checks by computing total energy and momentum

KE0 = sum(.5.*m.*v0.^2); %calculates the total kenetic energy initially
P0 = sum(m.*v0);         %calculates the total potential energy initially


% ----- Collision #1 -----

% *** carts 2 and 3 will collide

vA = [ 0 0 0 ];   % initialize vA

vA(1) = v0(1);  %no interaction - velocity of cart 1 remains constant in cm/s
vA(2) = (m(2)-m(3))/m23*v0(2) + (2*m(3)/m23)*v0(3); %resultant velocity of
                                                    %cart 2 in cm/s
vA(3) = 2*m(2)/m23*v0(2) - (m(2)-m(3))/m23*v0(3)  %resultant velocity of
                                                    %cart 3 in cm/s

% check energy and momentum (expectations)

KEA = sum(.5.*m.*vA.^2); %calculates the total kinetic energy of situation
checkKE_A = KEA-KE0 % Should be zero because no kenetic energy
                                    % is added or removed
PA = sum(m.*vA); %calculates the total potential energy of situation
checkP_A = PA-P0          % should be zero because no potential energy
                                    % added or removed

% check to see if there is another collision and output result

```

```
52      %   *** check BOTH pairs of adjacent carts, even if you know which is next
53      %   *** use IF/ELSE to output something simple yet meaningful.
54      %   *** end the IF/ELSE statement before starting the next collision
55
56  if vA(1) <= vA(2) && vA(3) >= vA(2) %checks if carts will never collide again
57      fprintf("There are no more collisions\n")
58  elseif vA(1) > vA(2) || vA(3) < vA(2) % checks if carts will collide
59      fprintf("There is another collision\n")
60  end
61
62
63  % ----- Collision #2 -----
64
65  % *** cart 1 and 2 will collide
66
67  vB = [ 0 0 0 ]; % initialize vB
68
69  vB(1) = (m(1)-m(2))/m12*vA(1) + (2*m(2))/m12*vA(2); %resultant velocity of
70                                                      %cart 1 in cm/s
71  vB(2) = (2*m(1)/m12)*vA(1) - (m(1)-m(2))/m12*vA(2); %resultant velocity of
72                                                      %cart 2 in cm/s
73  vB(3) = vA(3) %no interaction - velocity of cart 3 remains constant in cm/s
74
75  % check energy and momentum (expectations)
76
77  KEB = sum(.5.*m.*vB.^2);%calculates the total kinetic energy of situation
78  checkKE_B = KEB-KE0 % Should be zero because no kenetic energy
79                                          % is added or removed
80  PB = sum(m.*vB); %calculates the total potential energy of situation
81  checkP_B = PB-P0          % should be zero because no potential energy
82                                          % added or removed
83
84  % check to see if there is another collision and output result
85
86  if vB(1) <= vB(2) && vB(3) >= vB(2) %checks if carts will never collide again
87      fprintf("There are no more collisions\n")
88  elseif vB(1) > vB(2) || vB(3) < vB(2) % checks if carts will collide
89      fprintf("There is another collision\n")
90  end
91
92
93  % ----- Collision #3 -----
94
95  % *** carts 2 and 3 will collide
96
97  vC = [ 0 0 0 ];    % initialize vC
98
99  vC(1) = vB(1); %no interaction - velocity of cart 1 remains constant in cm/s
100 vC(2) = (m(2)-m(3))/m23*vB(2) + (2*m(3)/m23)*vB(3); %resultant velocity of
101                                                     %cart 2 in cm/s
102 vC(3) = (2*m(2)/m23)*vB(2) - (m(2)-m(3))/m23*vB(3)  %resultant velocity of
```

```
103                                                     %cart 3 in cm/s
104
105 % check energy and momentum (expectations)
106
107 KEC = sum(.5.*m.*vC.^2);%calculates the total kinetic energy of situation
108 checkKE_C = KEC-KE0       % Should be zero because no kenetic energy
109                                     % is added or removed
110 PC = sum(m.*vC); %calculates the total potential energy of situation
111 checkP_C = PC-P0          % should be zero because no potential energy
112                                     % added or removed
113
114 % check to see if there is another collision and output result
115
116 if vC(1) <= vC(2) && vC(3) >= vC(2) %checks if carts will never collide again
117     fprintf("There are no more collisions\n")
118 elseif vC(1) > vC(2) || vC(3) < vC(2) % checks if carts will collide
119     fprintf("There is another collision\n")
120 end
121
122 % ----- Collision #4 -----
123
124 % *** cart 1 and 2 will collide
125
126 vD = [ 0 0 0 ]; % initialize vD
127
128 vD(1) = (m(1)-m(2))/m12*vC(1) + (2*m(2))/m12*vC(2); %resultant velocity of
129                                                     %cart 1 in cm/s
130 vD(2) = (2*m(1)/m12)*vC(1) - (m(1)-m(2))/m12*vC(2); %resultant velocity of
131                                                     %cart 2 in cm/s
132 vD(3) = vC(3) %no interaction - velocity of cart 3 remains constant in cm/s
133
134 % check energy and momentum (expectations)
135
136 KED = sum(.5.*m.*vD.^2);%calculates the total kinetic energy of situation
137 checkKE_D = KED-KE0 % Should be zero because no kenetic energy
138                                     % is added or removed
139 PD = sum(m.*vD); %calculates the total potential energy of situation
140 checkP_D = PD-P0          % should be zero because no potential energy
141                                     % added or removed
142
143 % check to see if there is another collision and output result
144
145 if vD(1) <= vD(2) && vD(3) >= vD(2) %checks if carts will never collide again
146     fprintf("There are no more collisions\n")
147 elseif vD(1) > vD(2) || vD(3) < vD(2) % checks if carts will collide
148     fprintf("There is another collision\n")
149 end
150
151 % ----- Collision #5 -----
152
153 % *** carts 2 and 3 will collide
```

```
154
155 vE = [ 0 0 0 ];   % initialize vE
156
157 vE(1) = vD(1); %no interaction - velocity of cart 1 remains constant in cm/s
158 vE(2) = (m(2)-m(3))/m23*vD(2) + (2*m(3)/m23)*vD(3); %resultant velocity of
159                                                      %cart 2 in cm/s
160 vE(3) = (2*m(2)/m23)*vD(2) - (m(2)-m(3))/m23*vD(3)  %resultant velocity of
161                                                      %cart 3 in cm/s
162
163 % check energy and momentum (expectations)
164
165 KEE = sum(.5.*m.*vE.^2);%calculates the total kinetic energy of situation
166 checkKE_E = KEE-KE0       % Should be zero because no kenetic energy
167                                    % is added or removed
168 PE = sum(m.*vE); %calculates the total potential energy of situation
169 checkP_E = PE-P0          % should be zero because no potential energy
170                                    % added or removed
171
172 % check to see if there is another collision and output result
173
174 if vE(1) <= vE(2) && vE(3) >= vE(2) %checks if carts will never collide again
175     fprintf("There are no more collisions\n")
176 elseif vE(1) > vE(2) || vE(3) < vE(2) % checks if carts will collide
177     fprintf("There is another collision\n")
178 end
179
180 % *** Keep adding similar code until there are no more collisions
181
182 % *** There are 5 collisions before no more are possible
```

```
>> M3template_Fa23

m =

   300    60   240


v0 =

    36     9   -45


vA =

   36.0000  -77.4000  -23.4000


checkKE_A =

   5.8208e-11


checkP_A =

     0

There is another collision

vB =

   -1.8000  111.6000  -23.4000


checkKE_B =

     0


checkP_B =

     0

There is another collision

vC =

   -1.8000 -104.4000   30.6000


checkKE_C =
```

```
     0


checkP_C =

   9.0949e-13

There is another collision

vD =

  -36.0000   66.6000   30.6000


checkKE_D =

     0


checkP_D =

     0

There is another collision

vE =

  -36.0000    9.0000   45.0000


checkKE_E =

     0


checkP_E =

     0

There are no more collisions
>>
```