Call: avgPct ← markResult(18, 23, 19)

Givens: score1, score2, score3 (scores out of 25)
Results: avgPct (average of scores, out of 100)
Intermediates:
  sum (sum of scores)
  avgOutOf25 (average of scores, out of 25)

Header: avgPct ← markResult( score1, score2, score3 )

Body:
1. sum ← score1 + score2 + score3
2. avgOutOf25 ← sum / 3
3. avgPct ← avgOutOf25 * 4

Trace Table for avgPct ← markResult(18, 23, 19)

<table>
<thead>
<tr>
<th>Statement</th>
<th>score1</th>
<th>score2</th>
<th>score3</th>
<th>sum</th>
<th>avgOutOf25</th>
<th>avgPct</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial values</td>
<td>18</td>
<td>23</td>
<td>19</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>1. sum ← score1 + score2 + score3</td>
<td></td>
<td></td>
<td></td>
<td>60</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2. avgOutOf25 ← sum / 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>3. avgPct ← avgOutOf25 * 4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>80</td>
<td></td>
</tr>
</tbody>
</table>
Exercise 4-2 Tracing a Call

Program Memory

Givens: none
Results: none
Intermediates:
- **first**, **second**, **third** (three scores)
- **average** (average of scores, out of 100)

Header: **main()**

Body:
1. **printLine**("Please enter three scores")
2. **first** ← **readReal**()
3. **second** ← **readReal**()
4. **third** ← **readReal**
   (Call the MarkUser algorithm)
5. **average** ← **markResult**(first, second, third)
   (Print the average for the user)
6. **printLine**("The average is ", average)

Working memory

<table>
<thead>
<tr>
<th>first</th>
<th>second</th>
<th>third</th>
<th>average</th>
</tr>
</thead>
<tbody>
<tr>
<td>23</td>
<td>16</td>
<td>21</td>
<td>80</td>
</tr>
</tbody>
</table>

CPU

Givens: **score1**, **score2**, **score3** (scores out of 25)
Results: **avgPct** (average of scores, out of 100)
Intermediates: **sum** (sum of scores)
- **avgOutOf25** (average of scores, out of 25)

Header: **avgPct** ← **markResult**(score1, score2, score3)

Body:
1. **sum** ← score1 + score2 + score3
2. **avgOutOf25** ← sum / 3
3. **avgPct** ← avgOutOf25 * 4

Terminal Window

Please enter three scores
23 16 21
The average is 80
Table 1 - Table for main algorithm:

Interaction with user:
Please enter three scores out of 25
23 16 21
The average is 80 percent

<table>
<thead>
<tr>
<th>Statements</th>
<th>first</th>
<th>second</th>
<th>third</th>
<th>average</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial values</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>1. printLine(&quot;Please enter three scores&quot;)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2. first ← readReal()</td>
<td>23</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3. second ← readReal ()</td>
<td></td>
<td>16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4. third ← readReal ()</td>
<td></td>
<td></td>
<td>21</td>
<td></td>
</tr>
<tr>
<td>5. Call average ← markResult(first, second, third) (See Table 2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6. printLine(&quot;The average is &quot;, average)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Call algorithm markResult:

\[
\text{average} \leftarrow \text{markResult} (\text{first, second, third})\]
\[
\text{avgPct} \leftarrow \text{markResult} (\text{score1, score2, score3})
\]

Table 2 – Trace for \text{avgPct} \leftarrow \text{markResult} (23,16,21)

<table>
<thead>
<tr>
<th>Statement</th>
<th>score1</th>
<th>score2</th>
<th>score3</th>
<th>sum</th>
<th>avgOutOf25</th>
<th>avgPct</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial values</td>
<td>23</td>
<td>16</td>
<td>21</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>1. sum ← score1 + score2 + score3</td>
<td></td>
<td></td>
<td></td>
<td>60</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2. avgOutOf25 ← sum / 3</td>
<td></td>
<td></td>
<td></td>
<td>20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3. avgPct ← avgOutOf25 * 4</td>
<td></td>
<td></td>
<td></td>
<td>80</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Program Memory

Exercise 4-3 - Marks out of 100, again

Givens: score1, score2, score3 (scores out of 25)
Results: avgPct (average of scores, out of 100)
Intermediates:
   avgOutOf25 (average of scores, out of 25)
Header: avgPct ← markResult( score1, score2, score3 )
Body:
   1. avgOutOf25 ← average(score1, score2, score3)
   2. avgPct ← avgOutOf25 * 4

GIVENS: num1, num2, num3 (three numbers)
RESULTS: avg (the average of num1, num2, and num3)
HEADER: avg ← average(num1, num2, num3)
BODY:
   1. avg ← (num1 + num2 + num3)/3
Call to average ← markResult(23, 16, 21)

Givens: score1, score2, score3 (scores out of 25)
Results: avgPct (average of scores, out of 100)
Intermediates:
  avgOutOf25 (average of scores, out of 25)

Header: avgPct ← markResult(score1, score2, score3)

Body:
1. avgOutOf25 ← average(score1, score2, score3)
2. avgPct ← avgOutOf25 * 4

GIVENS: num1, num2, num3 (three numbers)
RESULTS: avg (the average of num1, num2, and num3)
HEADER: avg ← average(num1, num2, num3)
BODY:
1. avg ← (num1 + num2 + num3)/3
Table 1 - Table for avgPct ← markResult(23, 16, 21)

<table>
<thead>
<tr>
<th>Statements</th>
<th>score1</th>
<th>score2</th>
<th>score3</th>
<th>avgOutOf25</th>
<th>avgPct</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial values</td>
<td>23</td>
<td>16</td>
<td>21</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>1. Call avgOutOf25 ← average(23, 16, 21) (See Table 2)</td>
<td></td>
<td></td>
<td></td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>2. avgPct ← avgOutOf25 * 4</td>
<td></td>
<td></td>
<td></td>
<td>80</td>
<td></td>
</tr>
</tbody>
</table>

Call algorithm average:

avgOutOf25 ← average (score1, score2, score3)

avg ← average (num1, num2, num3)

Table 2 - Table for avg ← average(23, 16, 21)

<table>
<thead>
<tr>
<th>Statement</th>
<th>num1</th>
<th>num2</th>
<th>num3</th>
<th>avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial values</td>
<td>23</td>
<td>16</td>
<td>21</td>
<td>?</td>
</tr>
<tr>
<td>1. sum ← (num1 + num2 + num3)/3</td>
<td></td>
<td></td>
<td></td>
<td>20</td>
</tr>
</tbody>
</table>
GIVENS: n (a two digit number)
RESULTS: reverseN  (Same digits as n with reverse order)
INTERMEDIATES:
    tens, ones  (n’s left and right digit)
HEADER: reverseN ← rev2(n)
BODY:
1. (tens, ones) ← digits(n)
2. reverseN ← 10*ones + tens

The following algorithm is available to extract the ten’s and one’s digits from a two digit number:

(high, low) ← digits(x)
Trace for N = 42, i.e. reverseN ← rev2(42)

GIVENS: n (a two digit number)
RESULTS: reverseN (Same digits as n with reverse order)
INTERMEDIATES:
   tens, ones (n’s left and right digit)
HEADER: reverseN ← rev2(n)
BODY:
   3. (tens, ones) ← digits(n)
   4. reverseN ← 10*ones + tens

The following algorithm is available to extract the ten’s and one’s digits from a two digit number:
   (high, low) ← digits( X )

<table>
<thead>
<tr>
<th>Statements</th>
<th>n</th>
<th>tens</th>
<th>ones</th>
<th>reverseN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial values</td>
<td>42</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>1. Call digits(n)</td>
<td>4</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2. reverseN ← ones * 10 + tens</td>
<td></td>
<td></td>
<td>24</td>
<td></td>
</tr>
</tbody>
</table>

Call to (tens, ones) ← digits (n)
   (tens, ones) ← digits (n)
   (high, low) ← digits (x)
Givens: \( w, x, y, z \) (positive integers)
Result: \( \text{allJoined} \) (the result of joining \( w,x,y, \) and \( z \))
Intermediates:
\( wx \) (the results of joining \( w \) and \( x \))
\( yz \) (the result of joining \( y \) and \( z \))
Header: \( \text{allJoined} \leftarrow \text{join4}(w,x,y,z) \)
Body:
\[ \begin{align*}
wx & \leftarrow \text{join}(w,x) \\
yz & \leftarrow \text{join}(y,z) \\
\text{allJoined} & \leftarrow \text{join} (wx, yz)
\end{align*} \]

You may assume there is available an algorithm:
\[ \begin{align*}
c & \leftarrow \text{join}(a, b) \\
\text{Givens:} & \quad a, b, \text{two positive integers} \\
\text{Result:} & \quad c \text{ is the number having the digits} \\
& \qquad \text{in} \ a \text{ followed by the digits in} \ b.
\end{align*} \]