

Programming the TI-83/84 Plus RAM/RAMC

Note Title

2014-08-08

In Calculus, we learn that partitioning the area under a curve, gets an approximation for the area under the curve. As we use more partitions, we get closer to the exact area. The graphing calculators find answers very close to exact, so we need to program something less exact.

This program performs LRAM, MRAM, and RRAM for a specified number of partitions. You need to define the function in Y_1 .

```
0 PROGRAM:RAM
1 :Input "N PARTS
  ",N
2 :Input "LEFT END
  ",A
3 :Input "RIGHT EN
  D ",B
4 :(B-A)/N→W
5 :sum(W*seq(Y1,X,
  A,B-W,W))→T
6 :Disp "LRAM ",T
7 :sum(W*seq(Y1,X,
  A+W/2,B-W/2,W))→
  U
8 :Disp "MRAM ",U
9 :sum(W*seq(Y1,X,
  A+W,B,W))→V
10 :Disp "RRAM ",V
  :
```

```
0 PRGM → [ENTER] [X] MATH [÷] [ENTER]
1 PRGM → [1] [2nd] [ALPHA] [+] [LOG] [0] [8] MATH [X] [4] LN [0] [+]
  [ALPHA] ['] [ALPHA] [LOG] [ENTER]
2 PRGM → [1] [2nd] [ALPHA] [+] [)] SIN COS [4] [0] SIN LOG [x-1] [0]
  [+] [ALPHA] ['] [ALPHA] MATH [ENTER]
3 PRGM → [1] [2nd] [ALPHA] [+] [X] [x2] TAN [^] [4] [0] SIN LOG [x-1]
  [0] [+] [ALPHA] ['] [ALPHA] APPS [ENTER]
4 ( [ALPHA] APPS [-] [ALPHA] MATH [)] [÷] [ALPHA] LOG [STO→] [ALPHA] [-] [ENTER]
5 [2nd] STAT [↑] [↓] [5] [ALPHA] [-] [X] [2nd] STAT [↓] [5] VARS [↑] [ENTER]
  [ENTER] ['] [ALPHA] [STO→] ['] [ALPHA] MATH ['] [ALPHA] APPS [-] [ALPHA] [-] [']
  [ALPHA] [-] [)] [STO→] [ALPHA] [4] [ENTER]
6 PRGM → [3] [2nd] [ALPHA] [+] [)] [X] MATH [÷] [0] [+] [ALPHA] ['] [ALPHA]
  [4] [ENTER]
7 [2nd] STAT [↑] [↓] [5] [ALPHA] [-] [X] [2nd] STAT [↓] [5] VARS [↑] [ENTER]
  [ENTER] ['] [ALPHA] [STO→] ['] [ALPHA] MATH [+] [ALPHA] [-] [÷] [2] ['] [ALPHA]
  APPS [-] [ALPHA] [-] [÷] [2] ['] [ALPHA] [-] [)] [STO→] [ALPHA] [5]
  [ENTER]
8 PRGM → [3] [2nd] [ALPHA] [+] [÷] [X] MATH [÷] [0] [+] [ALPHA] ['] [ALPHA]
  [5] [ENTER]
9 [2nd] STAT [↑] [↓] [5] [ALPHA] [-] [X] [2nd] STAT [↓] [5] VARS [↑] [ENTER]
  [ENTER] ['] [ALPHA] [STO→] ['] [ALPHA] MATH [+] [ALPHA] [-] ['] [ALPHA] APPS [']
  [ALPHA] [-] [)] [STO→] [ALPHA] [6] [ENTER]
10 PRGM → [3] [2nd] [ALPHA] [+] [X] [X] MATH [÷] [0] [+] [ALPHA] ['] [ALPHA]
  [6] [ENTER]
```

```

Plot1 Plot2 Plot3
Y1=.25X^2+2
Y2=
Y3=
Y4=
Y5=
Y6=
Y7=
N PARTS 1
LEFT END 0
RIGHT END 4
LRAM
MRAM 8
RRAM 12
Done 24

```

```

Plot1 Plot2 Plot3
Y1=X^2
Y2=
Y3=
Y4=
Y5=
Y6=
Y7=
PRGMRAM
N PARTS 4
LEFT END 0
RIGHT END 4
LRAM
MRAM 14
RRAM 21
Done 30

```

In real-life applications, we don't always have an explicit function. Sometimes, we just get a table of data. So, we can use RAML - RAM List.

To use this program, enter your list into L1. For this function, we don't use N; we use W - the partition width.

```

0 PROGRAM:RAML
1 :Input "W ",W
2 :dim(L1)→N
3 :sum(L1)→S
4 :Disp "LRAM ",(S
5 -L1(N))*W
6 :Disp "MRAM ",(2
  *S-L1(1)-L1(N))*
  W/2
  :Disp "RRAM ",(S
  -L1(1))*W
  :

```



Length of a Road You and a companion are driving along a twisty stretch of dirt road in a car whose speedometer works but whose odometer (mileage counter) is broken. To find out how long this particular stretch of road is, you record the car's velocity at 10-sec intervals, with the results shown in the table below. (The velocity was converted from mi/h to ft/sec using $30 \text{ mi/h} = 44 \text{ ft/sec}$.) Estimate the length of the road by averaging the LRAM and RRAM sums.

Time (sec)	Velocity (ft/sec)	Time (sec)	Velocity (ft/sec)
0	0	70	15
10	44	80	22
20	15	90	35
30	35	100	44
40	30	110	30
50	44	120	35
60	35		

L1	L2	L3	1
0	-----	-----	
44			
15			
30			
30			
44			
35			

L1(1) = 0

L1	L2	L3	1
15			
22			
35			
44			
30			
35			

L1(14) =

```

prgmRAML
W 10
LRAM
MRAM 3490
RRAM 3665
3840
Done

```