Includes easy-entry keystroke listings.

HEWLETT-PACKARD

HP-75

USERS' LIBRARY SOLUTIONS

Math II

NOTICE

The program material contained herein is supplied without representation or warranty of any kind. Hewlett-Packard Company therefore assumes no responsibility and shall have no liability, consequential or otherwise, of any kind arising from the use of this program material or any part thereof.

TABLE OF CONTENTS

TABLE OF CONTENTS (continued)

PROGRAM DESCRIPTION

MIDPOINT RULE FOR INTEGRATION

This program approximates the integral of the continuous function f

$$\int_a^b f(x)\,dx$$

by the Riemann sum

$$\sum f(x_i)h$$

in which h=(b-a)/n and the points x_i are the midpoints of the subintervals [a+(i-1)h, a+ih], i=1,...,n.

f, the function, is supplied at line 80.

a, b, and n are entered by the user when requested.

The calculation is expressed as follows:

h=(b-a)/n	calculate step size
m=0	initialize accumulator
FOR x=a+h/2 to b-h/2 step h	
m=m+f(x)	accumulate functional values at mid-points
NEXT x	
m=mh	multiply by width of interval
display m	display approximation

SAMPLE PROBLEM

A) Estimate $\int_0^1 (\mathrm{SIN}(x)/x)\,dx$ with 100 and 1000 subintervals

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
A_1	Enter integrand in line 80	80 DEF FNF(X)=SIN(X)/X	
2	Run program	***MIDPOINT RULE***	
3	Enter lower limit	Lower limit of integral?	0 [RTN]
4	Enter upper limit	Upper limit of integral?	1 [RTN]
5	Enter number of subintervals	Number of subintervals?	100 [RTN]
6	Answer	Mdpt approx =.946084325239	[RTN]
7	Run again	Run again, View again, or End? R	[RTN]
8	Enter lower limit	Lower limit of integral?	0 [RTN]
9	Enter upper limit	Upper limit of integral?	1 [RTN]
10	Enter new number of subintervals	Number of subintervals?	1000 [RTN]
11	Answer	Mdpt approx =.946083082905	[RTN]
12	End	Run again, View again, or End?	E [RTN]

SAMPLE PROBLEM

B) Estimate $\int_0^1 EXP(-x^2)\,dx$ with 16 and 100 subintervals

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
B 1	Enter integrand in line 80	80 DEF FNF(X)=EXP(-X^2)	
2	Run program	***MIDPOINT RULE***	
3	Enter lower limit	Lower limit of integral?	0 [RTN]
4	Enter upper limit	Upper limit of integral?	1 [RTN]
5	Enter number of subintervals	Number of subintervals?	16 [RTN]
6	Answer	Mdpt approx =.746943912519	[RTN]
7	Run again	Run again, View again, or End? R	[RTN]
8	Enter lower limit	Lower limit of integral?	0 [RTN]
9	Enter upper limit	Upper limit of integral?	1 [RTN]
10	Enter new number of subintervals	Number of subintervals?	100 [RTN]
11	Answer	Mdpt approx =.746827198498	[RTN]
12	End	Run again, View again, or End?	E [RTN]

SAMPLE PROBLEM

C) Estimate $\int_0^1 \text{SIN}(x^2)\,dx$ with 100 and 1000 subintervals

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
C 1	Enter integrand in line 80	80 DEF FNF(X)=SIN(X^2)	
2	Run program	***MIDPOINT RULE***	
3	Enter lower limit	Lower limit of integral?	0 [RTN]
4	Enter upper limit	Upper limit of integral?	1 [RTN]
5	Enter number of subintervals	Number of subintervals?	100 [RTN]
6	Answer	Mdpt approx =.310263799031	[RTN]
7	Run again	Run again, View again, or End? R	[RTN]
8	Enter lower limit	Lower limit of integral?	0 [RTN]
9	Enter upper limit	Upper limit of integral?	1 [RTN]
10	Enter new number of subintervals	Number of subintervals?	1000 [RTN]
11	Answer	Mdpt approx =.310268256706	[RTN]
12	End	Run again, View again, or End?	E [RTN]

USER INSTRUCTIONS

STEP	INSTRUCTIONS	DISPLAY	INPUT
1	Place the integrand, as a	80 DEF FNF(X)= user function	
	user defined function in line 80.		
2	Run program	***MIDPOINT RULE***	
3	Enter lower limit	Lower limit of integral?	a [RTN]
4	Enter upper limit	Upper limit of integral?	b [RTN]
5	Enter number of subintervals	Number of subintervals?	n [RTN]
6	Answer	Mdpt approx =	[RTN]
7	Repeat options	Run again, View again, or End? R	R [RTN]
7a	To run again, press [RTN]		or
7b	To view the answer again,		V [RTN]
	enter V [RTN]		or
7c	To end the program		E [RTN]
	enter E [RTN]		

VARIABLE NAMES

NAME	DESCRIPTION	NAME	DESCRIPTION
FNF(X)	Integrand	H	Length of subinterval
A	Lower limit of integration	M	Accumulator for sum of functional evaluations at midpoints and final approximations.
B	Upper limit of integration		
N	Number of subintervals	Q$	Test string for Run again, View again, or End.

NOTES AND REFERENCES

References: Conte, S.D., and deBoor, C., ELEMENTARY NUMERICAL ANALYSIS (McGraw-Hill, New York, 1980).

PROGRAM LISTING

```
 10 ! MIDPOINT RULE
 20 ! Approximate an integral
 30 ! over a finite interval
 40 ! using the midpoint rule
 50 ! Revision 11/01/82
 60 DISP '   * * * MIDPOINT RULE * * *'
 70 ! Define integrand
 80 DEF FNF(X) = SIN(X^2)                    -User defined function
 90 ! Enter limits, # of subint
100 INPUT 'Lower limit of integral?';A
110 INPUT 'Upper limit of integral?';B
120 IF A=B THEN BEEP @ DISP 'Invalid ra
    nge for integration!' @ GOTO 100
130 INPUT 'Number of sub-intervals?';N
140 DISP 'Computing...'
150 ! Compute increment, sum=0
160 H=(B-A)/N                               -Calculate the step size
170 M=0                                     -Initialize accumulator
180 ! Main loop
190 FOR X=A+H/2 TO B-H/2 STEP H
200 M=M+FNF(X)                              -Accumulate midpoint sums
210 NEXT X
220 M=M*H                                   -Multiply by width of interval
230 DISP 'Mdpt approx =';M                  -Display approximation
240 ! Wait for return key
250 IF NUM(KEY$)#13 THEN 250 ELSE 270
260 ! Run,view, or end
270 DISP CHR$(210);'un again, ';CHR$(21     -Continuation options
    4);'iew again, or ';CHR$(197);
280 INPUT 'nd?','R'; Q$ @ Q$=UPRC$(Q$[1
    ,1])
290 ON POS('RVE',Q$)+1 GOTO 270,100,230
    ,300
300 STOP
```

PROGRAM DESCRIPTION

TRAPEZOIDAL RULE FOR INTEGRATION

This program approximates the integral of the continuous function f

$$\int_a^b f(x)\,dx$$

by sums of the form

$$\sum (f(x_i)+f(x_{i+1}))h/2$$

in which $h=(b-a)/n$, the points $x_i=a+ih$, and $i=0,\ldots,n$.

f, the function, is supplied at line 90.

a,b, and n are supplied by the user on request.

The calculation is expressed as follows:

```
h=(b-a)/n                    calculate the increment
t=(f(a)+f(b))/2              initialize trapezoidal approximation
FOR x=a+h to b-h step h
  t=t+f(x)                   accumulate function results
NEXT x
t=th                         calculate final approximation
display t
```

SAMPLE PROBLEM

A) Estimate $\int_0^1 x^2 dx$ using 37 and 64 subintervals

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
A 1	Enter integrand in 90	90 DEF FNF(X)=X^2	
2	Run program	***TRAPEZOIDAL RULE***	
3	Enter lower limit	Lower limit of integral?	0 [RTN]
4	Enter upper limit	Upper limit of integral?	1 [RTN]
5	Enter number of subintervals	Number of subintervals?	37 [RTN]
6	Answer	Trap approx =.333455076697	[RTN]
7	Run again	Run again, View again, or End? R	[RTN]
8	Enter lower limit	Lower limit of integral?	0 [RTN]
9	Enter upper limit	Upper limit of integral?	1 [RTN]
10	Enter new subintervals	Number of subintervals	64 [RTN]
11	Answer	Trap approx =.333374023436	[RTN]
12	End	Run again, View again, or End?	E [RTN]

SAMPLE PROBLEM

B) Estimate $\int_0^1 (2/(2+\text{SIN}(10\pi x)))dx$ using 45 and 64 subintervals

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
B 1	Enter integrand in 90	90 DEF FNF(X)=2/(2+SIN(10*PI*X))	
2	Run program	***TRAPEZOIDAL RULE***	
3	Enter lower limit	Lower limit of integral?	0 [RTN]
4	Enter upper limit	Upper limit of integral?	1 [RTN]
5	Enter number of subintervals	Number of subintervals?	45 [RTN]
6	Answer	Trap approx =1.15470053826	[RTN]
7	Run again	Run again, View again, or End? R	[RTN]
8	Enter lower limit	Lower limit of integral?	0 [RTN]
9	Enter upper limit	Upper limit of integral?	1 [RTN]
10	Enter new subintervals	Number of subintervals?	64 [RTN]
11	Answer	Trap approx =1.15470053838	[RTN]
12	End	Run again, View again, or End?	E [RTN]

SAMPLE PROBLEM

C) Estimate $\int_0^1 \sqrt{SIN(x)}dx$ using 100 and 1000 subintervals

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
C 1	Enter integrand in line 90	90 DEF FNF(X)=SQR(SIN(X))	
2	Run program	***TRAPEZOIDAL RULE***	
3	Enter lower limit	Lower limit of integral?	0 [RTN]
4	Enter upper limit	Upper limit of integral?	1 [RTN]
5	Enter number of subintervals	Number of subintervals?	100 [RTN]
6	Answer	Trap approx =.642772202535	[RTN]
7	Run again	Run again, View again, or End? R	[RTN]
8	Enter lower limit	Lower limit of integral?	0 [RTN]
9	Enter upper limit	Upper limit of integral?	1 [RTN]
10	Enter new subintervals	Number of subintervals?	1000 [RTN]
11	Answer	Trap approx =.642971085268	[RTN]
12	End	Run again, View again, or End? R	E [RTN]

USER INSTRUCTIONS

STEP	INSTRUCTIONS	DISPLAY	INPUT
1	Place the integrand as a user defined function in line 90	90 DEF FNF(X)= users function	
2	Run program	***TRAPEZOIDAL RULE***	
3	Lower limit	Lower limit of integral?	a [RTN]
4	Upper limit	Upper limit of integral?	b [RTN]
5	Enter number of subintervals	Number of subintervals?	n [RTN]
6	Answer (delays until [RTN])	Trap approx =	[RTN]
7	Repeat options	Run again, View again, or End? R	R [RTN]
7a	To run again, press [RTN]		or
7b	To view the answer again,		V [RTN]
	enter V [RTN]		or
7c	To end the program		E [RTN]
	enter E [RTN]		

VARIABLE NAMES

NAME	DESCRIPTION	NAME	DESCRIPTION
FNF(X)	Integrand	H	Length of subinterval
A	Lower limit of integration	T	Accumulator for weighted functional sums and trapezoidal approximation
B	Upper limit of integration		
N	Number of subintervals	Q$	Test input string for Run again, View answer again or End.

NOTES AND REFERENCES

References: Conte, S.D., and deBoor, C., ELEMENTARY NUMERICAL ANALYSIS, (McGraw-Hill, New York, 1980).

PROGRAM LISTING

```
 10 ! TRAPEZOIDAL RULE
 20 ! approximate the integral
 30 ! over a finite interval
 40 ! using the area of
 50 ! trapezoids
 60 ! Revision 11/01/82
 70 DISP '  * * * TRAPEZOIDAL RULE * *
    *'
 80 ! Define integrand
 90 DEF FNF(X) = SQR(SIN(X))                  -User defined function
100 ! Enter limits,# of subint
110 INPUT 'Lower limit of integral?';A
120 INPUT 'Upper limit of integral?';B
130 IF A=B THEN BEEP @ DISP 'Invalid ra
    nge for integration!' @ GOTO 110
140 INPUT 'Number of subintervals?';N
150 DISP 'Computing...'
160 ! Compute increment,edpt contrib
170 T=(FNF(A)+FNF(B))/2
180 H=(B-A)/N                                -Compute step size
190 ! Main loop
200 FOR X=A+H TO B-H STEP H
210 T=T+FNF(X)                               -Accumulate sums
220 NEXT X
230 T=T*H                                    -Compute final approximation
240 DISP 'Trap approx=';T                    -Display approximation
250 ! Wait for return key to continue
260 IF NUM(KEY$)#13 THEN 260 ELSE 280
270 ! Run,view,or end                        -Continuation options
280 DISP CHR$(210);'un again,';CHR$(214
    );'iew again,or ';CHR$(197);
290 INPUT 'nd?','R'; Q$ @ Q$=UPRC$(Q$[1
    ,1])
300 ON POS('RVE',Q$)+1 GOTO 280,110,240
    ,310
310 STOP
```

PROGRAM DESCRIPTION

ROMBERG RULE FOR INTEGRATION

This program speeds up the convergence of the trapezoidal rule for approximating the integral

$$\int_a^b f(x)\,dx$$

and it is based on successive bisections of the interval [a,b]. The recursion formula for the k'th trapezoidal sum is:

t(k)=(t(k-1) + m(k))/2

in which m(k) is the k'th midpoint sum. The extrapolated sequence is defined by:

$$t(i,k)=t(i,k-1)+(t(i,k-1)-t(i-1,k-1))/(4^k-1)$$

for i and k non-negative and in which t(0,k)=t(k). The sequence u(i,k) defined by:

u(i,k)=2t(i+1,k)-t(i,k)

lies on the opposite side of the limit from t(i,k) and helps to end the computation.

f, the function, is supplied at line 140.

a and b are entered when requested.

The calculation is expressed as follows:

h=b-a	calculate interval
t(0)=(f(a)+f(b)) h/2	calculate t(0)
FOR k=1 to 16	loop for 16 sums
s=0, h=h/2	initialize sum, divide increment
q=1	initialize extrapolation factor

PROGRAM DESCRIPTION

ROMBERG RULE FOR INTEGRATION (continued)

```
      FOR x=a+h to b-h step 2h
        s=s+f(x)                          accumulate midpoint sums
      NEXT x
      t(k)=t(k-1)/2+hs                    calculate kth sum
      FOR i=k-1 to 0 step -1
        q=4q
        u(i)=2t(i+1)-t(i)                 calculate stopping sums
        if |t(i)-u(i)| < 1E-10 then (a)
        t(i) = t(i+1) + (t(i+1)-t(i))/(q-1)  calculate extrapolation sum
      NEXT i
    NEXT k
(a) display final sum t(0)
```

SAMPLE PROBLEM

A) Estimate $\int_0^1 \text{SIN}(x^2)dx$

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
A 1	Enter integrand in line 140	140 DEF FNF(X)=SIN(X^2)	
2	Run program	***ROMBERG INTEGRATION***	
3	Enter lower limit	Lower limit of integral?	0 [RTN]
4	Enter upper limit	Upper limit of integral?	1 [RTN]
5		Roll through t(0,0),...,t(4,6)	
6	Answer	Final approx =.310268301722	[BACK]
7	View last row of extrapolation	t(0,6) =.310290287875	[RTN]
		t(1,6) =.310268296948	[RTN]
		t(2,6) =.310268301727	[RTN]
		t(3,6) =.310268301724	[RTN]
		t(4,6) =.310268301724	[RTN]
		t(5,6) =.310268301722	[RTN]
8	Answer	Final approx =.310268301722	[RTN]
9	End	Run again, View again, or End? R	E [RTN]

SAMPLE PROBLEM

B) Estimate $\int_0^1 EXP(-x^2)dx$

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
B 1	Enter integrand in line 140	140 DEF FNF(X)=EXP(-X^2)	
2	Run program	***ROMBERG INTEGRATION***	
3	Enter lower limit	Lower limit of integral?	0 [RTN]
4	Enter upper limit	Upper limit of integral?	1 [RTN]
5		Roll through t(0,0),...,t(3,6)	
6	Answer	Final approx =.746824132813	[BACK]
7	View last row of extrapolation	t(0,6) =.746809163638	[RTN]
		t(1,6) =.746824133299	[RTN]
		t(2,6) =.746824132812	[RTN]
		t(3,6) =.746824132812	[RTN]
		t(4,6) =.746824132813	[RTN]
		t(5,6) =.746824132814	[RTN]
8	Answer	Final approx =.746824132813	[RTN]
9	End	Run again, View again, or End? R	E [RTN]

SAMPLE PROBLEM

C) Estimate $\int_0^1 (2/(2+SIN(10\pi x)))dx$

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
C 1	Enter integrand in line 140	140 DEF FNF(X)=2/(2+SIN(10*PI*X))	
2	Run program	***ROMBERG INTEGRATION***	
3	Enter lower limit	Lower limit of integral?	0 [RTN]
4	Enter upper limit	Upper limit of integral?	1 [RTN]
5		Rolls through t(0,0),...,t(0,7)	
6	Answer	Final approx =1.15470053838	[RTN]
7	End	Run again, View again, or End? R	E [RTN]

SAMPLE PROBLEM

D) Estimate $\int_0^1 (1/(1+EXP(x)))dx$

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
D 1	Enter integrand in line 140	140 DEF FNF(X)=1/(1+EXP(X))	
2	Run program	***ROMBERG INTEGRATION***	
3	Enter lower limit	Lower limit of integral?	0 [RTN]
4	Enter upper limit	Upper limit of integral?	1 [RTN]
5		Rolls through t(0,0),...,t(3,5)	
6	Answer	Final approx =.379885493042	[RTN]
7	End	Run again, View again, or End? R	E [RTN]

USER INSTRUCTIONS

STEP	INSTRUCTIONS	DISPLAY	INPUT
1	Place the integrand as a user defined function in line 140	140 DEF FNF(X)= users function	
2	Run program	***ROMBERG INTEGRATION***	
3	Enter lower limit	Lower limit of integral?	a [RTN]
4	Enter upper limit	Upper limit of integral?	b [RTN]
5	Answer	Final approx =	
6	Review last extrapolated row	t(i,j) =	[BACK] to start review
			[RTN] goes forward
7	Repeat options	Run again, View again, or End?	R [RTN]
7a	To run again, press [RTN]		or
7b	To view the answer again,		V [RTN]
	enter V [RTN]		or
7c	To end the program		E [RTN]
	enter E [RTN]		

VARIABLE NAMES

NAME	DESCRIPTION	NAME	DESCRIPTION
FNF(X)	Integrand	T(I)	Trapezoidal, extrapolated sums
FNA(X)	Answer function	U(I)	Stopping sums
A	Lower limit of integration	S	Midpoint sums
B	Upper limit of integration	Q	Extrapolation factor
H	Length of subinterval, increment	Q$	Control string for RVE module

NOTES AND REFERENCES

References: Stoer, J.D., and Bulirsch, R., INTRODUCTION TO NUMERICAL ANALYSIS, (Springer-Verlay, New York, 1981).

PROGRAM LISTING

```
 10 ! ROMBERG
 20 ! approximates the value of
 30 ! the integral over a finite
 40 ! interval using extrap.
 50 ! to the limit
 60 ! of trapezoidal sums
 70 ! revision 11/01/1982
 80 DISP '  * * ROMBERG INTEGRATION * *
    '
 90 ! Define answer function
100 DEF FNA
110 A=NUM(KEY$) @ IF A<13 AND A<>8 THEN
    110
120 FNA=A @ END DEF
130 ! Define integrand
140 DEF FNF(X) = 1/(1+EXP(X))
150 ! Enter limits of integral
160 INPUT 'Lower limit of integral?';A
170 INPUT 'Upper limit of integral?';B
180 IF A=B THEN BEEP @ DISP 'Invalid ra
    nge for integration!' @ GOTO 160
190 ! Arrays for summing and stopping
200 DIM T(16)
210 DIM U(16)
220 ! Compute 0'th trap sum
230 H=B-A @ T(0)=H/2*(FNF(A)+FNF(B))      -Compute interval, T(0)
240 DISP 't( 0 , 0 )=';T(0)
250 ! Set main loop and subloop for tra
    p sums
260 FOR K=1 TO 16                        -Loop for 16 sums
270 S=0 @ H=H/2                          -Initialize sum, divide
                                          increment
280 Q=1                                  -Initialize extrapolation
                                          factor
290 FOR X=A+H TO B-H STEP 2*H
300 S=S+FNF(X)                           -Accumulate midpoint sums
310 NEXT X
320 T(K)=T(K-1)/2+S*H                    -Compute Kth sum
330 DISP 't( 0 ,';K;')=';T(K)
340 ! Subloop for extrapolations
350 FOR I=K-1 TO 0 STEP -1
360 Q=4*Q
370 ! Stopping criterion
380 U(I)=T(I+1)*2-T(I)                   -Compute stopping sums
390 IF ABS(T(I)-U(I))<=.00000000001 THE
    N 450
400 ! Computation of extrapolations
410 T(1)=T(I+1)+1/(Q-1)*(T(I+1)-T(I))    -Compute extrapolation sums
420 DISP 't(';K-I;',';K;')=';T(I)
430 NEXT I
440 NEXT K
450 DISP 'Final approx=';T(I)            -Display final approximation
460 ! Review last row of extrap          -Pressing BACK shows the last
                                          row
```

PROGRAM LISTING

```
470 IF FNA=8 THEN 480 ELSE 550
480 FOR J=0 TO K-1
490 DISP 't(';J;',';K;')=';T(K-J)
500 IF FNA=8 THEN J=MAX(J-1,0) @ GOTO 4
    90
510 NEXT J
520 IF FNA=8 THEN 480
530 DISP 'Final approx=';T(I) @ IF FNA=
    8 THEN 480
540 ! Run,view, or end                    -Continuation options
550 DISP CHR$(210);'un again,';CHR$(214
    );'iew again,or ';CHR$(197);
560 INPUT 'nd?','R'; Q$ @ Q$=UPRC$(Q$[1
    ,1])
570 ON POS('RVE',Q$)+1 GOTO 550,160,480
    ,580
580 STOP
```

PROGRAM DESCRIPTION

SIMPSON'S RULE FOR INTEGRATION

This program approximates the integral of the continuous function f

$$\int_a^b f(x)\ dx$$

by the sum over n=2k equal subintervals

$$\sum((f(x_{i-1})+f(x_i))/2+f(y_i))h$$

in which $h=(b-a)/k$, $x_i=a+ih$, $y_i=a+(i-\frac{1}{2})\ h$ and i=1,...,k in the sum.

f, the function, is supplied at 130.

a,b, and n are supplied by the user on request.

The calculation is expressed as follows:

```
h=2(b-a)/n                      calculate modified increment

m,t=0                           initialize midpoint and trapezoidal
                                        approximation

FOR x=a+h/2 to b-h/2 step h

  m=m+f(x)                      accumulate midpoint approximation

  t=t+(f(x-h/2)+f(x+h/2))/2     accumulate trapezoidal approximation

NEXT x

m=mh, t=th                      complete midpoint and trapezoidal
                                        approximation

s=(t+2m)/3                      calculate Simpson's approximation

display s
```

SAMPLE PROBLEM

A) estimate $\int_0^1 \text{SIN}(x^2)dx$ using 38 subintervals

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
A 1	Enter integrand in 130	130 DEF FNF(X)=SIN(X^2)	
2	Run program	***SIMPSON'S RULE***	
3	Enter lower limit	Lower limit of integral?	0 [RTN]
4	Enter upper limit	Upper limit of integral?	1 [RTN]
5	Enter number of subintervals	Number of subintervals?	38 [RTN]
6	Answer	Simpson approx =.310268263282	[RTN]
7	Answer	Mdpt approx =.310143443667	[RTN]
8	Answer	Trap approx =.310517902512	[RTN]
9	End	Run again, View again, or End? R	E [RTN]

SAMPLE PROBLEM

B) estimate $\int_0^1 EXP(x^2)dx$ using 128 subintervals

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
B_1	Enter integrand in 130	130 DEF FNF(X)=EXP(X^2)	
2	Run program	***SIMPSON'S RULE***	
3	Enter lower limit	Lower limit of integral?	0 [RTN]
4	Enter upper limit	Upper limit of integral?	1 [RTN]
5	Enter number of subintervals	Number of subintervals?	128 [RTN]
6	Answer	Simpson approx = 1.46265174704	[RTN]
7	Answer	Mdpt approx = 1.46259644627	[RTN]
8	Answer	Trap approx = 1.46276234858	[RTN]
9	End	Run again, View again, or End?	E [RTN]

SAMPLE PROBLEM

C) estimate $\int_0^1 (x^2+5x+1)dx$ using 8 subintervals

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
C 1	Enter integrand in 130	130 DEF FNF(X)=X^2+5*X+1	
2	Run program	***SIMPSON'S RULE***	
3	Enter lower limit	Lower limit of integral?	1 [RTN]
4	Enter upper limit	Upper limit of integral?	2 [RTN]
5	Enter number of subintervals	Number of subintervals	8 [RTN]
6	Answer	Simpson approx =10.8333333333	[RTN]
7	Answer	Mdpt approx =10.82825	[RTN]
8	Answer	Trap approx =10.84375	[RTN]
9	End	Run again, View again, or End? R	E [RTN]

USER INSTRUCTIONS

STEP	INSTRUCTIONS	DISPLAY	INPUT
1	Place the integrand as a user defined function in line 130	130 DEF FNF(X)= users function	
2	Run program	***SIMPSON'S RULE***	
3	Enter lower limit	Lower limit of integral?	a [RTN]
4	Enter upper limit	Upper limit of integral?	b [RTN]
5	Enter number of subintervals	Number of subintervals?	n [RTN]
6	Answer	Simpson approx =	[RTN]
7	Compare with midpoint approx	Mdpt approx =	[RTN]/[BACK]
8	Compare with trapezoidal approx	Trap approx =	[RTN]/[BACK]
9	Repeat options	Run again, View again, or End? R	R [RTN]
9a	To run again, press [RTN]		or
9b	To view the answer again,		V [RTN]
	enter V [RTN]		or
9c	To end the program		E [RTN]
	enter E [RTN]		

VARIABLE NAMES

NAME	DESCRIPTION	NAME	DESCRIPTION
FNF(X)	Integrand	H	Modified increment
FNA(X)	Test function for control	M	Midpoint approximation
A	Lower limit of integration	T	Trapezoidal approximation
B	Upper limit of integration	S	Simpson approximation
N	Number of subintervals	Q$	Test string for control

NOTES AND REFERENCES

References: Conte, S.D., and deBoor, C., ELEMENTARY NUMERICAL ANALYSIS, (McGraw-Hill, New York, 1980).

PROGRAM LISTING

```
 10 ! SIMPSON'S RULE
 20 ! Approximate the value of
 30 ! the integral over a finite
 40 ! interval using the area
 50 ! under a parabola
 60 ! Revision 11/01/82
 70 DISP '   * * * SIMPSONS RULE * * *'
 80 ! Wait for return key
 90 DEF FNA(X)
100 A=NUM(KEY$) @ IF A#13 AND A#8 THEN
    100
110 FNA=A @ END DEF
120 ! Define integrand
130 DEF FNF(X) = X^2+5*X+1                  -User defined function
140 ! Enter limits,# of subint.
150 INPUT 'Lower limit of integral?';A
160 INPUT 'Upper limit of integral?';B
170 IF A=B THEN BEEP @ DISP 'Invalid ra
    nge for integration!' @ GOTO 150
180 INPUT 'Number of subintervals?';N
190 IF MOD(N,2)=1 THEN BEEP @ DISP 'Cho
    ose even number of subint.' @ GOTO
    180
200 DISP 'Computing...'
210 ! Compute increment and initialize
220 H=2/N*(B-A)                             -Compute modified increment
230 M=0 @ T=0                               -Initialize midpoint and trap
                                             approximations
240 ! Main loop
250 FOR X=A+H/2 TO B-H/2 STEP H
260 M=M+FNF(X)                              -Accumulate midpoint
                                             approximation
270 T=T+1/2*(FNF(X-H/2)+FNF(X+H/2))         -Accumulate trapezoidal
                                             approximation
280 NEXT X
290 T=T*H @ M=M*H                           -Complete mdpt, trap
                                             approximations
300 S=1/3*(T+2*M)                           -Compute Simpsons approximation
310 ! Answer display
320 DISP 'Simpson approx=';S                -Display approximations
330 A=FNA(0)
340 DISP 'Mdpt approx=';M @ A=FNA(0) @
    IF A=8 THEN 320
350 DISP 'Trap approx=';T @ A=FNA(0) @
    IF A=8 THEN 340
360 ! Run,view,or end
370 DISP CHR$(210);'un again, ';CHR$(21  -Continuation options
    4);'iew again,or ';CHR$(197);
380 INPUT 'nd ?','R'; Q$ @ Q$=UPRC$(Q$[
    1,1])
390 ON POS('RVE',Q$)+1 GOTO 370,150,320
    ,400
400 STOP
```

PROGRAM DESCRIPTION

NEWTON-COTES RULE FOR INTEGRATION

This program approximates the integral of the continuous function f

$$\int_a^b f(x)\,dx$$

by sums of the form

$$\sum s(i)\ f(x_{k,i})\Delta \text{ in which } \Delta=h/(840(n+1)),\ x_{k,i}=a+\frac{(6k+i)}{6(n+1)}\,h,\ h=b-a.$$

The sum is extended over $0\le i\le 6$, $0\le k\le n$ and the s(i) are predetermined weights.

f, the function to be integrated, is supplied at line 90.

a,b,n are requested and entered by the user.

The calculation is expressed as follows:

```
h=b-a                                  calculate the interval h
t=0                                    initialize the integrand t
FOR k=0 to n                           calculate approximation at k intervals
  c=0                                  initialize the integrand for the interval
  FOR i=0 to 6                         calculate approximation at kth interval
    c=c+s(i) f(a+(6k+i)h/6(n+1))
  NEXT i
  c=c h/840(n+1)                       complete calculation of integral
  t=t+c                                accumulate integral t
NEXT k
display t
```

SAMPLE PROBLEM

A) Estimate $\int_0^1 EXP(-x^2)dx$ with 12 and 64 subintervals.

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
A 1	Enter integrand in line 90	90 DEF FNF(X)=EXP(-X^2)	
2	Run program	**NEWTON-COTES FORMULA**	
3	Enter lower limit	Lower limit of integral?	0 [RTN]
4	Enter upper limit	Upper limit of integral?	1 [RTN]
5	Enter number of subintervals	Number of subintervals?	12 [RTN]
6	Answer	N-C approx =.746824132814	[RTN]
7	Run again	Run again, View again, or End? R	[RTN]
8	Enter lower limit	Lower limit of integral?	0 [RTN]
9	Enter upper limit	Upper limit of integral?	1 [RTN]
10	Enter new number of subintervals	Number of subintervals?	64 [RTN]
11	Answer	N-C approx =.746824132818	[RTN]
12	End	Run again, View again, or End? R	E [RTN]

SAMPLE PROBLEM

B) Estimate $\int_0^1 SQR(1-x^2)dx = \int_0^1 (1-x^2)^{\frac{1}{2}}dx$ with 37 and 64 subintervals.

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
B 1	Enter integrand in line 90	90 DEF FNF(X)=SQR(1-X^2)	
2	Run program	**NEWTON-COTES FORMULA**	
3	Enter lower limit	Lower limit of integral?	0 [RTN]
4	Enter upper limit	Upper limit of integral?	1 [RTN]
5	Enter number of subintervals	Number of subintervals?	37 [RTN]
6	Answer	N-C approx =.78537177313	[RTN]
7	Run again	Run again, View again, or End? R	[RTN]
8	Enter lower limit	Lower limit of integral?	0 [RTN]
9	Enter upper limit	Upper limit of integral?	1 [RTN]
10	Enter new number of subintervals	Number of subintervals?	64 [RTN]
11	Answer	N-C approx =.785386367651	[RTN]
12	End	Run again, View again, or End? R	E [RTN]

SAMPLE PROBLEM

C) Estimate $\int_0^1 \mathrm{SIN}(x^2)dx$ with 13 and 37 subintervals.

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
C 1	Enter integrand in line 90	90 DEF FNF(X)=SIN(X^2)	
2	Run program	**NEWTON-COTES FORMULA**	
3	Enter lower limit	Lower limit of integral?	0 [RTN]
4	Enter upper limit	Upper limit of integral?	1 [RTN]
5	Enter number of subintervals	Number of subintervals?	13 [RTN]
6	Answer	N-C approx =.310268301724	[RTN]
7	Repeat	Run again, View again, or End? R	[RTN]
8	Enter lower limit	Lower limit of integral?	0 [RTN]
9	Enter upper limit	Upper limit of integral?	1 [RTN]
10	Enter new number of subintervals	Number of subintervals?	37 [RTN]
11	Answer	N-C approx =.310268301725	[RTN]
12	End	Run again, View again, or End? R	E [RTN]

USER INSTRUCTIONS

STEP	INSTRUCTIONS	DISPLAY	INPUT
1	Place integrand as a user defined function in line 90	90 DEF FNF(X)= users function	
2	Run program	**NEWTON-COTES FORMULA**	
3	Enter lower limit	Lower limit of integral?	a [RTN]
4	Enter upper limit	Upper limit of integral?	b [RTN]
5	Enter number of subintervals	Number of subintervals?	n [RTN]
6	Answer	N-C approx =	[RTN]
7	Repeat options	Run again, View again, or End?	R [RTN]
7a	To run again, press [RTN]		or
7b	To view the answer again,		V [RTN]
	enter V [RTN]		or
7c	To end the program		E [RTN]
	enter E [RTN]		

VARIABLE NAMES

NAME	DESCRIPTION	NAME	DESCRIPTION
FNF(X)	Integrand	S(I)	Weights i=1,...,6
A	Lower limit of integration	C	Accumulates sums on subintervals
B	Upper limit of integration	T	Accumulates sums on entire interval
N	Number of subintervals	Q$	Control string for RVE module
H	Length of interval		

NOTES AND REFERENCES

References: Stoer, J., and Bulirsch, R., INTRODUCTION TO NUMERICAL ANALYSIS, (Springer-Verlag, New York, 1980).

PROGRAM LISTING

```
 10 ! NEWTON-COTES
 20 ! Approximates the value of
 30 ! the integral over a finite
 40 ! interval using the area
 50 ! under a higher order curve
 60 ! Revision 11/01/82
 70 DISP '  * * NEWTON-COTES FORMULA *
    *'
 80 ! Define integrand
 90 DEF FNF(X) = SIN(X^2)                  -User defined function
100 ! Store limits,number of sub-
110 ! intervals
120 INPUT 'Lower limit of integral?';A
130 INPUT 'Upper limit of integral?';B
140 IF A=B THEN BEEP @ DISP 'Invalid ra    -Check limits of integration
    nge for integration!' @ GOTO 120
150 INPUT 'Number of subintervals?';N
160 DISP 'Computing...'
170 ! Store weights
180 S(0)=41 @ S(1)=216 @ S(2)=27 @ S(3)    -Initialize weights
    =272 @ S(4)=27 @ S(5)=216 @ S(6)=41
190 ! Compute length of interval
200 ! and initialize
210 H=B-A @ T=0                            -Compute width of interval,
                                            initialize integrand
220 ! Main loops
230 FOR K=0 TO N                           -Compute approx. at K intervals
240 C=0                                    -Initialize the integrand for
                                            the interval
250 FOR I=0 TO 6                           -Compute the approximation at
                                            the Kth interval
260 C=C+S(I)*FNF(A+H*(6*K+1)/(6*(N+1)))
270 NEXT I
280 C=C*H/((N+1)*840)                      -Complete computation of the
                                            integral
290 T=T+C
300 NEXT K
310 DISP 'N-C approx=';T                   -Display approximation
320 ! Wait for return key to continue
330 IF NUM(KEY$)#13 THEN 330 ELSE 350
340 ! Run,view, or end                     -Continuation options
350 DISP CHR$(210);'un again,';CHR$(214
    );'iew again, or ';CHR$(197);
360 INPUT 'nd ?','R'; Q$ @ Q$=UPRC$(Q$[
    1,1])
370 ON POS('RVE',Q$)+1 GOTO 350,120,310
    ,380
380 STOP
```

PROGRAM DESCRIPTION

EULER'S METHOD

Euler's method provides a numerical approximation to the initial value problem

$$x'=dx/dt=f(x,t),\ x(a)=x_o$$

in the neighborhood of the point t=a where the boundary value

$$x(a)=x_o$$

is assigned.

x(t) is expanded using a first degree Taylor polynomial about the point t

$$x(t+h) = x(t) + h\,x'(t) + O(h^2)$$
$$= x(t) + h\,f(x,t) + O(h^2)$$

where h is the step size h=(b-a)/n and b is the desired point of solution.

The program computes the series of iterates

$$x_{i+1}=x_i+hf(x_i,t_i),\ \text{ in which } t_i=a+ih,\ i=0,\ldots,\ n-1,$$

with x_n the final approximation at t=b, and h=(b-a)/n.

f, the function to be integrated, is supplied at line 100.

a,x_o,b,n are entered by the user when requested.

The calculation is expressed as follows:

h=(b-a)/n	calculate step size
FOR i=0 to n-1	calculate approx. at n intervals
$y=x_o+h\,f(x_o,a+ih)$	calculate approx. for the ith interval.
$x_o=y$	update x_o
NEXT i	
Display y	

SAMPLE PROBLEM

A) Approximate the value of the solution, with initial condition $x(0)=.5$, to the differential equation $dx/dt=1+(x-t)^2$, at $t=1$.

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
A 1	Enter f(x,t) in line 100	100 DEF FNF(X,T)=1+(X-T)^2	
2	Run program	***EULERS METHOD***	
3	Enter initial t-value	Initial t-value?	0 [RTN]
4	Enter initial x-value	Initial x-value?	.5 [RTN]
5	Enter endpoint	Endpoint for solution?	1 [RTN]
6	Step size	Number of subint, a to b?	10 [RTN]
7	Answer	Approx solution =1.94220484185	[RTN]
8	Run again	Run again, View again, or End? R	[RTN]
9	Enter initial t-value	Initial t-value?	0 [RTN]
10	Enter initial x-value	Initial x-value?	.5 [RTN]
11	Enter endpoint	Endpoint for solution?	1 [RTN]
12	Enter new subintervals	Number of subint, a to b?	40 [RTN]
13	Answer	Approx solution =1.98351090675	[RTN]
14	End	Run again, View again, or End?	E [RTN]

SAMPLE PROBLEM

B) Approximate the value of the solution, with initial condition $x(0)=0$, to the differential equation $dx/dt=1/(1+x^2+t^2)$, at $t=1$.

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
B 1	Enter f(x,t) in line 100	100 DEF FNF(X,T)=1/(1+X^2+T^2)	
2	Run program	***EULERS METHOD***	
3	Enter initial t-value	Initial t-value?	0 [RTN]
4	Enter initial x-value	Initial x-value?	0 [RTN]
5	Enter endpoint	Endpoint for solution?	1 [RTN]
6	Step size	Number of subint, a to b?	10 [RTN]
7	Answer	Approx solution =.730071796614	[RTN]
8	Repeat	Run again, View again, or End? R	[RTN]
9	Enter initial t-value	Initial t-value?	0 [RTN]
10	Enter initial x-value	Initial x-value?	0 [RTN]
11	Enter endpoint	Endpoint of solution?	1 [RTN]
12	Enter new subintervals	Number of subint, a to b?	100 [RTN]
13	Answer	Approx solution =.707070075028	[RTN]
14	End	Run again, View again, or End? R	E [RTN]

SAMPLE PROBLEM

C) Approximate the value of the solution, with initial condition x(0)=0, to the differential equation $dx/dt=t-x^2$, at t=1.

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
C 1	Enter f(x,t) in line 100	100 DEF FNF(X,T)=T-X^2	
2	Run program	***EULERS METHOD***	
3	Enter initial t-value	Initial t-value?	0 [RTN]
4	Enter initial x-value	Initial x-value?	0 [RTN]
5	Enter endpoint	Endpoint for solution?	1 [RTN]
6	Step size	Number of subint, a to b?	16 [RTN]
7	Answer	Approx solution =.435594383877	[RTN]
8	Repeat	Run again, View again, or End?	[RTN]
9	Enter initial t-value	Initial t-value?	0 [RTN]
10	Enter initial x-value	Initial x-value?	0 [RTN]
11	Enter endpoint	Endpoint for solution?	1 [RTN]
12	Enter new subintervals	Number of subint, a to b?	32 [RTN]
13	Answer	Approx solution =.445780315035	[RTN]
14	End	Run again, View again, or End? R	E [RTN]

USER INSTRUCTIONS

STEP	INSTRUCTIONS	DISPLAY	INPUT
1	Place integrand as user defined function in line 100	100 DEF FNF(X,T)= users function	
2	Run program	***EULERS METHOD***	
3	Enter initial t-value	Initial t-value?	a [RTN]
4	Enter initial x-value	Initial x-value?	Xo [RTN]
5	Enter endpoint	Endpoint for solution?	b [RTN]
6	Step size	Number of subint, a to b?	n [RTN]
7	Answer	Approx solution =	[RTN]
8	Repeat options	Run again, View again, or End?	R [RTN]
8a	To run again, press [RTN]		or
8b	To view the answer again,		V [RTN]
	enter V [RTN]		or
8c	To end the program		E [RTN]
	enter E [RTN]		

VARIABLE NAMES

NAME	DESCRIPTION	NAME	DESCRIPTION
FNF(X,T)	Integrand	H	Step size
A	Initial t-value	F	Temporary storage for initial x-value
XØ	Initial x-value, iterates of X	Y	Iterates of x, final approximation
B	Endpoint, point at which solution is to approx-imated.	Q$	Control string for repeat options
		N	Number of subintervals [a,b] is to be divided

NOTES AND REFERENCES

References: Braun, M., DIFFERENTIAL EQUATIONS AND THEIR APPLICATIONS, (Springer-Verlag, New York, 1978).

PROGRAM LISTING

```
 10 ! EULER'S METHOD
 20 ! approximate the value of a
 30 ! a solution to a differential
 40 ! equation of the first order
 50 ! using the first term
 60 ! of the Taylor expansion
 70 ! Revision 11/01/1982
 80 DISP '   * * * EULERS METHOD * * *'
 90 ! Define function to be integrated
100 DEF FNF(X,T) = -X^2+T                  -User defined function
110 ! Store initial value,endpoint
120 INPUT 'Initial t-value?';A
130 INPUT 'Initial x-value?';X0
140 Z=X0
150 INPUT 'Endpoint for solution?';B
160 IF A=B THEN BEEP @ DISP 'Invalid in
    terval for solution!' @ GOTO 120
170 ! Store number of subinter-
180 ! vals from a to b, com-
190 ! pute step size
200 INPUT 'Number of subint,a to b?';N
210 DISP 'Computing...'
220 H=(B-A)/N                              -Calculate step size
230 ! Main loop
240 FOR I=0 TO N-1
250 Y=X0+H*FNF(X0,A+I*H)                   -Calculate approximation at ith
                                            interval
260 X0=Y
270 NEXT I
280 DISP 'Approx solution=';Y              -Display approximation
290 ! Wait for return key to continue
300 IF NUM(KEY$)#13 THEN 300 ELSE 320
310 ! Run,view,or end
320 DISP CHR$(210);'un again,';CHR$(214    -Continuation options
    );'iew again, or ';CHR$(197);
330 INPUT 'nd ?','R'; Q$ @ Q$=UPRC$(Q$[
    1,1])
340 ON POS('RVE',Q$)+1 GOTO 320,120,280
    ,350
350 STOP
```

PROGRAM DESCRIPTION

NEWTON'S METHOD

This program finds the root of the equation f(x)=0 starting from an initial guess x=a. The root is found by the following iteration

$$x_{i+1} = x_i - f(x_i)/f'(x_i)$$

where the function f(x) is supplied at line 80 and the derivative f'(x)=g(x) is supplied at line 110. An initial guess for the root is supplied on request. The calculation is expressed as follows: initialize counter, iterates to prevent oscillation.

a = initial guess

c=1,a(i)=0 for i=0,1,2

y=a-f(a)/g(a)

c=c+1

If |a-y|<EPS then disp y, stop

a(MOD(c,3))=y

If |a(2)-a(0)|<EPS then disp y, stop

a=y

iterate again

[EPS= 1E-499 is the smallest available number]

The iteration converges to the root under the condition that $g(x) \neq 0$ and g(x) is monotonically increasing (or decreasing) in the region of interest.

SAMPLE PROBLEM

A) Calculate $\sqrt{2}$ as the root of $f(x)=x^2-2$. In this case $g(x)=f'(x)=2x$.

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
A 1	Enter f(x) in line 80	80 DEF FNF(X)=X^2-2	
2	Enter f'(x) in line 110	110 DEF FNG(X)=2*X	
3	Run program	***NEWTONS METHOD***	
4	Enter initial guess	Initial guess?	1 [RTN]
5	Answer	Root =1.41421356237	[RTN]
6	Repeat	Run again, View again, or End? R	E [RTN]

SAMPLE PROBLEM

B) Calculate the smallest positive root of f(x)=EXP(-x)-SIN(x).
Here g(x)=f'(x)=-EXP(-x)-COS(x).

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
B 1	Enter f(x) in line 80	80 DEF FNF(X)=EXP(-X)-SIN(X)	
2	Enter f'(x) in line 110	110 DEF FNG(X)=-EXP(-X)-COS(X)	
3	Run program	***NEWTONS METHOD***	
4	Enter initial guess	Initial guess?	1 [RTN]
5	Answer	Root =.588532743982	[RTN]
6	Repeat	Run again, View again, or End? R	E [RTN]

SAMPLE PROBLEM

C) Solve Kepler's equation E-e*SIN(E)=M, in which e=0.01672 and M=2.567126065. The root of f(x)=x-e*SIN(x)-M, will yield the desired solution. The derivative of f(x) is g(x)=1-e*COS(x).

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
C 1	Enter f(x), using the values for e,M, in line 80	80 DEF FNF(X)=X-0.01672*SIN(X) -2.567126065	
2	Enter f'(x), using the values for e,M, in line 110	110 DEF FNG(X)=1-0.01672*COS(X)	
3	Run program	***NEWTONS METHOD***	
4	Enter initial guess	Initial guess?	0 [RTN]
5	Answer	Root =2.57608537977	[RTN]
6	End	Run again, View again, or End? R	E [RTN]

SAMPLE PROBLEM

D) The polynomial $f(x)=x^3-6x^2-2x+12$ has three real roots. Find them! $f'(x)=3x^2-12x-2$.

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
D 1	Enter f(x) in line 80	80 DEF FNF(X)=X^3-6*X^2-2*X+12	
2	Enter f'(x) in line 110	110 DEF FNG(X)=3*X^2-12*X-2	
3	Run program	***NEWTONS METHOD***	
4	Enter initial guess	Initial guess?	0 [RTN]
5	Answer	Root =6	[RTN]
6	Run again	Run again, View again, or End? R	[RTN]
7	Enter initial guess	Initial guess?	1 [RTN]
8	Answer	Root =1.41421356238	[RTN]
9	Run again	Run again, View again, or End?	[RTN]
10	Enter initial guess	Initial guess?	-1 [RTN]
11	Answer	Root =-1.41421356237	[RTN]
12	Run again	Run again, View again, or End? R	E [RTN]

USER INSTRUCTIONS

STEP	INSTRUCTIONS	DISPLAY	INPUT
1	Enter f(x) in line 80 as user defined function	80 DEF FNF(X)= users function	
2	Enter f'(x) in line 110 as user defined function	110 DEF FNG(X)= derivative of users function	
3	Run program	***NEWTON'S METHOD***	
4	Enter initial guess	Initial guess?	A [RTN]
5	Answer	Root =	[RTN]
6	Repeat options	Run again, View again, or End?	R [RTN]
6a	To run again, press [RTN]		or
6b	To view the answer again,		V [RTN]
	enter V [RTN]		or
6c	To end the program		E [RTN]
	enter E [RTN]		

VARIABLE NAMES

NAME	DESCRIPTION	NAME	DESCRIPTION
FNF(X)	Function f(x) whose root(s) is (are) to be found	A	Initial guess
		Y	Iterate
FNG(X)	Derivative of f(x)=f'(x)	Q$	Control string for RVE module
A(I)	Mini-array, preserving last three iterates, i=0, 1, 2	C	Counter, used to set A(I)

NOTES AND REFERENCES

References: Henrici, P., ELEMENTS OF NUMERICAL ANALYSIS (John Wiley & Sons, Inc., New York, 1964).

Lax, R.S., Burstein, and A. Lax, CALCULUS WITH APPLICATIONS AND COMPUTING, (V. 1, Springer-Verlag, New York, 1976).

PROGRAM LISTING

```
 10 ! NEWTON'S METHOD
 20 ! Finds the roots of a
 30 ! function by using the fun-
 40 ! ction and its derivative
 50 DISP '   * * * NEWTONS METHOD * * *
    '
 60 ! Revision 11/01/82
 70 ! Define function
 80 DEF FNF(X) = EXP(-X)-SIN(X)          -User defined function
 90 ! Define the derivative of
100 ! the function
110 DEF FNG(X) = -EXP(-X)-COS(X)
120 ! Initialize counter and
130 ! stopping-check array
140 C=1 @ A(0)=0 @ A(1)=0 @ A(2)=0
150 ! Initial guess
160 INPUT 'Initial guess?';A
170 IF FNG(A)=0 THEN BEEP @ DISP 'Inval  -Check validity of guess
    id guess!' @ GOTO 160
180 ! Iterate and step counter
190 Y=A-FNF(A)/FNG(A)
200 C=C+1
210 DISP 'Current iterate=';Y
220 ! Stopping criterion and
230 ! check for oscillatory
240 ! behavior of approximat'n
250 IF ABS(A-Y)<EPS THEN 280            -Test for stopping
260 A(MOD(C,3))=Y
270 IF ABS(A(2)-A(0))<EPS THEN 280 ELSE  -Test for stopping
     A=Y @ GOTO 190
280 DISP 'Root=';Y                      -Display root
290 ! Wait for return key to continue
300 IF NUM(KEY$)#13 THEN 300 ELSE 310
310 ! Run,view,or end                   -Continuation options
320 DISP CHR$(210);'un again,';CHR$(214
    );'iew again,or';CHR$(197);
330 INPUT 'nd?','R'; Q$ @ Q$=UPRC$(Q$[1
    ,1])
340 ON POS('RVE',Q$)+1 GOTO 320,160,280
    ,350
350 STOP
```

PROGRAM DESCRIPTION

TRAPEZOIDAL RULE FOR ORDINARY DIFFERENTIAL EQUATIONS

This method is an advancement of Euler's method in solving the initial value problem

$$x'(t) = \frac{dx}{dt} = f(x,t)$$

in the neighborhood of t=a where the value $x(a) = x_o$ is assigned. The solution x(t) is expanded using a second order Taylor polynomial about the point t

$$x(t+h)=x(t)+hx'(t)+(h^2/2)x''(t)+O(h^3)$$

Substituting the differential equation and its derivative into this equation yields:

$$x(t+h)=x(t)+h(f(x,t)+(h/2)(f_t(x,t)+f_x(x,t)f(x,t)))+O(h^3)$$

The factor of h in this equation is approximated by

$$(f(x,t)+f(x+hf(x,t),t+h))/2 = g(x,t,h)$$

The program computes the series of iterates

$$x_{i+1}=x_i+hg(x_i,t_i,h)$$

in which $t_i=a+ih$, i=0,...,n-1, with x_n the final approximation at t=b, and h=(b-a)/n.

f, the function, is supplied at line 90.

a,x_o, b, and n are entered by the user when requested.

The calculation is expressed as follows:

```
h=(b-a)/n                          calculate step size
FOR i=0 to n-1                     calculate n sums
  y=x_o+h g(x_o, a+ih, h)
  x_o=y                            update x_o
NEXT i
display y
```

SAMPLE PROBLEM

A) Approximate the value of the solution, with initial condition x(2)=2, to the differential equation dx/dt=1-x/t, at t=2.1.

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
A 1	Enter f(x,t) in line 90	90 DEF FNF(X,T)=1-X/T	
2	Run program	**TRAPEZOIDAL RULE FOR ODE**	
3	Enter initial t-value	Initial t-value?	2 [RTN]
4	Enter initial x-value	Initial x-value?	2 [RTN]
5	Enter endpoint	Endpoint for solution?	2.1 [RTN]
6	Enter number of subintervals	Number of subintervals a to b?	8 [RTN]
7	Answer	Approx solution =2.00238095239	[RTN]
8	Repeat	Run again, View again, or End? R	[RTN]
9	Enter initial t-value	Initial t-value?	2 [RTN]
10	Enter initial x-value	Initial x-value?	2 [RTN]
11	Enter endpoint	Endpoint for solution?	2.1 [RTN]
12	Enter new subintervals	Number of subintervals a to b?	64 [RTN]
13	Answer	Approx solution =2.00238095239	[RTN]
14	End	Run again, View again, or End? R	E [RTN]

SAMPLE PROBLEM

B) Approximate the value of the solution, with initial condition x(0)=1, to the differential equation dx/dt=x, at t=1 and t=-1.

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
B 1	Enter f(x,t) in line 90	90 DEF FNF(X,T)=X	
2	Run program	***TRAPEZOIDAL RULE FOR ODE***	
3	Enter initial t-value	Initial t-value?	0 [RTN]
4	Enter initial x-value	Initial x-value?	1 [RTN]
5	Enter endpoint	Endpoint for solution?	1 [RTN]
6	Enter number of subintervals	Number of subintervals a to b?	32 [RTN]
7	Answer	Approx solution =2.71784967399	[RTN]
7a	Run again	Run again, View again, or End? R	[RTN]
8	Enter initial t-value	Initial t-value?	0 [RTN]
9	Enter initial x-value	Initial x-value?	1 [RTN]
10	Enter endpoint	Endpoint for solution?	-1 [RTN]
11	Enter new number of subintervals	Number of subintervals a to b?	64 [RTN]
12	Answer	Approx solution =.367894587051	[RTN]
13	End	Run again, View again, or End? R	[RTN]

SAMPLE PROBLEM

C) Approximate the value of the solution, with initial condition x(0)=0, to the differential equation $dx/dt=3t^2 + 1000(t^3-x)$ at t=1.

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
C 1	Enter f(x,t) in line 90	90 DEF FNF(X,T)=3*T^2+1000(T^3-X)	
2	Run program	***TRAPEZOIDAL RULE FOR ODE***	
3	Enter initial t-value	Initial t-value?	0 [RTN]
4	Enter initial x-value	Initial x-value?	0 [RTN]
5	Enter endpoint	Endpoint for solution?	1 [RTN]
6	Enter number of subintervals	Number of subintervals a to b?	100 [RTN]
7	Answer	Approx solution =2.78826181824E154	[RTN]
8	Repeat	Run again, View again, or End? R	[RTN]
9	Enter initial t-value	Initial t-value?	0 [RTN]
10	Enter initial x-value	Initial x-value?	0 [RTN]
11	Enter endpoint	Endpoint for solution?	1 [RTN]
12	Enter new numbers of subintervals	Number of subintervals a to b?	1000 [RTN]
13	Answer (takes a few minutes)	Approx solution =1.000002996	[RTN]
14	End	Run again, View again, or End? R	E [RTN]

USER INSTRUCTIONS

STEP	INSTRUCTIONS	DISPLAY	INPUT
1	Place integrand as a user defined function in line 90	90 DEF FNF(X,T)= users function	
2	Run program	***TRAPEZOIDAL RULE FOR ODE***	
3	Enter initial t-value	Initial t-value?	a [RTN]
4	Enter initial x-value	Initial x-value?	X_0 [RTN]
5	Enter endpoint	Endpoint for solution?	b [RTN]
6	Enter number of subintervals	Number of subintervals a to b?	n [RTN]
7	Answer	Approx solution =	[RTN]
8	Repeat options	Run again, View again, or End?	R [RTN]
8a	To run again, press [RTN]		or
8b	To view the answer again,		V [RTN]
	enter V [RTN]		or
8c	To end the program		E [RTN]
	enter E [RTN]		

VARIABLE NAMES

NAME	DESCRIPTION	NAME	DESCRIPTION
FNF(x,t)	Integrand	N	Number of subintervals into which [a,b] is to be subdivided
FNG(x,t,h)	Trapezoidal (approximation)		
A	Initial t-value	H	Step size
X_0	Initial x-value, iterated values	T	Temporary storage for original initial x-value
B	Endpoint or t-value where solution is to be approximated	Y	Iterates of x, final approximation
		Q$	Control string for Run, View or End

NOTES AND REFERENCES

References: Gear, C.W., NUMERICAL INITIAL VALUE PROBLEMS IN ORDINARY DIFFERENTIAL EQUATIONS, (Prentice-Hall, Englewood Cliffs, 1971).

PROGRAM LISTING

```
 10 ! TRAPEZOIDAL RULE FOR ODE
 20 ! Approximate the value of
 30 ! a solution to a first
 40 ! order differential equat'n
 50 ! using trapezoidal sums
 60 ! Revision 11/01/1982
 70 DISP '* * TRAPEZOIDAL RULE FOR ODE
    * *'
 80 ! Define function to be integrated
 90 DEF FNF(X,T) = 3*T^2+1000*(T^3-X)      -User defined function
100 ! Define trapezoidal approximation
    to f(x,t)
110 DEF FNG(X,T,H) = (FNF(X,T)+FNF(X+H*
    FNF(X,T),T+H))/2
120 ! Store initial value,endpoint
130 INPUT 'Initial t-value?';A
140 INPUT 'Initial x-value?';X0
150 Z=X0
160 INPUT 'Endpoint for solution?';B
170 IF A=B THEN BEEP @ DISP 'Invalid in
    terval for solution!' @ GOTO 130
180 ! Store number of subinter-
190 ! vals from a to b,compute
200 ! step size
210 INPUT 'Number of subintervals,a to
    b?';N
220 DISP 'Computing...'
230 H=(B-A)/N                             -Compute step size
240 ! Main loop
250 FOR I=0 TO N-1
260 Y=X0+H*FNG(X0,A+I*H,H)                -Accumulate approximation
270 X0=Y
280 NEXT I
290 DISP 'Approx solution=';Y             -Display approximation
300 ! Wait for return key to continue
310 IF NUM(KEY$)<>13 THEN 310 ELSE 330
320 ! Run,view, or end                    -Continuation options
330 DISP CHR$(210);'un again,';CHR$(214
    );'iew again, or ';CHR$(197);
340 INPUT 'nd?','R'; Q$ @ Q$=UPRC$(Q$[1
    ,1])
350 ON POS('RVE',Q$)+1 GOTO 330,130,290
    ,360
360 STOP
```

PROGRAM DESCRIPTION

RUNGE-KUTTA

The RUNGE-KUTTA program uses the fourth-order Runge-Kutta method to provide a numerical approximation to the solution x(t) of the ordinary differential equation

$$dx/dt=x'(t)=f(x,t)$$

in the neighborhood of the point t=a where the boundary value

$$x(a)=x_o$$

is specified. To determine the value of x at t=b, the program computes a series of values x_m at n intermediate points t_m:

$$x_{m+1} = x_m + hg(x_m,t_m,h)$$

where

$$h \equiv (b - a)/n$$

$$t_m \equiv a + mh \qquad m = 0,1,\ldots,n-1$$

The increment function g is defined by

$$g(x_m,t_m,h) = (k_1 + 2k_2 + 2k_3 + k_4)/6$$

$$k_1 = f(x_m,t_m)$$

$$k_2 = f(x_m+hk_1/2,t_m+h/2)$$

$$k_3 = f(x_m+hk_2/2,t_m+h/2)$$

$$k_4 = f(x_m+hk_3,t_m+h)$$

The user enters the function f as a BASIC statement function in line 90 of the program. The program will halt during execution to prompt the user to imput values for a,b,x_o, and n. Note that in general, the accuracy of the result increases with smaller step size, i.e., with increasing n, but execution time also increases proportionately. The display will show the value of each successive x_m as it is computed.

SAMPLE PROBLEM

A) Approximate the value of the solution, with initial condition $x(0)=.5$ to the differential equation $dx/dt=1+(x-t)^2$, at $t=1$.

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
A 1	Enter f(x,t) in 90	90 DEF FNF(X,T)=1+(X-T)^2	
2	Run program	***RUNGE-KUTTA METHOD***	
3	Enter initial t-value	Initial t-value?	0 [RTN]
4	Enter initial x-value	Initial x-value?	.5 [RTN]
5	Enter end point	End point for solution?	1 [RTN]
6	Step size	Number of subintervals, a to b?	20 [RTN]
7	Answer	Approx solution=1.9999999244	[RTN]
8	Repeat	Run again, View again, or End? R	[RTN]
9	Enter initial t-value	Initial t-value?	0 [RTN]
10	Enter initial x-value	Initial x-value?	.5 [RTN]
11	Enter end point	End point for solution?	1 [RTN]
12	Enter new step size	Number of subintervals a to b?	100 [RTN]
13	Answer	Approx solution=1.99999999991	[RTN]
14	End	Run again, View again, or End? R	E [RTN]

SAMPLE PROBLEM

B) Approximate the value of the solution, with initial condition $x(1)=-1$, to the differential equation $dx/dt=1/t^2-x/t-x^2$, at $t=2$.

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
B 1	Enter f(x,t) in 90	90 DEF FNF(X)=1/T^2-X/T-X^2	
2	Run program	***RUNGE-KUTTA METHOD***	
3	Enter initial t-value	Initial t-value?	1 [RTN]
4	Enter initial x-value	Initial x-value?	-1 [RTN]
5	Enter endpoint	Endpoint for solution?	2 [RTN]
6	Step size	Number of subintervals a to b?	32 [RTN]
7	Answer	Approx solution =-.499999961648	[RTN]
8	Repeat	Run again, View again, or End? R	R [RTN]
9	Enter initial t-value	Initial t-value?	1 [RTN]
10	Enter initial x-value	Initial x-value?	-1 [RTN]
11	Enter endpoint	Endpoint for solution?	2 [RTN]
12	New step size	Number of subintervals a to b?	128 [RTN]
13	Answer	Approx solution =-.499999999852	[RTN]
14	End	Run again, View again, or End?	E [RTN]

SAMPLE PROBLEM

C) Approximate the value of the solution, with initial condition $x(-1)=.009900990099$. ($\overline{.0099}=1/101$), to the differential equation $dx/dt=-200tx^2$, at $t=0$.

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
C 1	Enter f(x,t) in line 90	90 DEF FNF(X,T)=-200*T*X^2	
2	Run program	***RUNGE-KUTTA METHOD***	
3	Enter initial t-value	Initial t-value?	-1 [RTN]
4	Enter initial x-value	Initial x-value?	.009900990099 [RTN]
5	Enter endpoint	Endpoint for solution?	0 [RTN]
6	Step size	Number of subintervals a to b?	100 [RTN]
7	Answer	Approx solution =.999972322634	[RTN]
8	Repeat	Run again, View again, or End? R	R [RTN]
9	Enter initial t-value	Initial t-value?	-1 [RTN]
10	Enter initial x-value	Initial x-value?	$\overline{0099}$ [RTN]
11	Enter endpoint	Endpoint for solution?	0 [RTN]
12	Enter new step size	Number of subintervals a to b?	1000 [RTN]
13	Answer	Approx solution =.999999996471	[RTN]
14	End	Run again, View again, or End? R	E [RTN]

USER INSTRUCTIONS

STEP	INSTRUCTIONS	DISPLAY	INPUT
1	Place integrand as a user defined function in line 90	90 DEF FNF(X,T)= users function	
2	Run program	***RUNGE-KUTTA METHOD***	
3	Enter initial t-value	Initial t-value?	a [RTN]
4	Enter initial x-value	Initial x-value?	X_0 [RTN]
5	Enter endpoint	Endpoint for solution?	b [RTN]
6	Step size	Number of subintervals a to b?	n [RTN]
7	Answer	Approx solution =	[RTN]
8	Repeat options	Run again, View again, or End? R	R [RTN]
8a	To run again, press [RTN]		or
8b	To view the answer again,		V [RTN]
	enter V [RTN]		or
8c	To end the program		E [RTN]
	enter E [RTN]		

VARIABLE NAMES

NAME	DESCRIPTION	NAME	DESCRIPTION
FNF(X,T)	Integrand	H	Step size
K_1, K_2 K_3, K_4	Weighted functional evaluations of f(x,t)	T	Temporary storage for original initial x-value
		U	Temporary independent variable to simplify formulas
A	Initial t-value	B	Endpoint, final point of evaluation for solution
Q$	Control string for Run again, View again, or End	Y	Iterates of X, final approximation
N	Number of subintervals into which [a,b] is subdivided	X_0	Initial x-value, iterated values

NOTES AND REFERENCES

References: Conte, S.D., and deBoor, C., ELEMENTARY NUMERICAL ANALYSIS, (McGraw Hill, New York, 1980).

Stoer, J., and Bulirsch, R., INTRODUCTION TO NUMERICAL ANALYSIS, (Springer-Verlag, New York, 1980).

PROGRAM LISTING

```
 10 ! RUNGE-KUTTA METHOD
 20 ! Approximate the value of a
 30 ! a solution to a first
 40 ! order differential equat'n
 50 ! using parabolic sums
 60 ! Revision 11/01/1982
 70 DISP ' * * * RUNGE-KUTTA METHOD * * *'
 80 ! Define function to be integrated
 90 DEF FNF(X,T) = -200*T*X^2                    -User defined function
100 ! Store initial value,endpoint
110 INPUT 'Initial t-value?';A
120 INPUT 'Initial x-value?';X0
130 Z=X0
140 INPUT 'Endpoint for solution?';B
150 IF A=B THEN BEEP @ DISP 'Invalid interval for solution!' @ GOTO 110
160 ! Store number of subinter-
170 ! vals from a to b,compute
180 ! step size
190 INPUT 'Number of subintervals,a to b?';N
200 DISP 'Computing...'
210 H=(B-A)/N                                   -Compute step size
220 ! Main loop
230 FOR I=0 TO N-1
240 U=A+I*H
250 K1=FNF(X0,U)                                -Calculate weighted evaluations
                                                 of f(x,t)
260 K2=FNF(X0+H*K1/2,U+H/2)
270 K3=FNF(X0+H*K2/2,U+H/2)
280 K4=FNF(X0+H*K3,U+H)
290 Y=X0+H*(K1+2*K2+2*K3+K4)/6                  -Accumulate approximation
300 X0=Y
310 NEXT I
320 DISP 'Approx solution=';Y                   -Display approximation
330 ! Wait for return key to continue
340 IF NUM(KEY$)#13 THEN 340 ELSE 360
350 ! Run,view, or end                          -Continuation options
360 DISP CHR$(210);'un again,';CHR$(214);'iew again, or ';CHR$(197);
370 INPUT 'nd?','R'; Q$ @ Q$=UPRC$(Q$[1,1])
380 ON POS('RVE',Q$)+1 GOTO 360,110,320,390
390 STOP
```

PROGRAM DESCRIPTION

CONTRACTION MAPPING

A function f which has the property that there exists a positive number s, $s<1$, for which $|f(x)-f(y)|\leq s|x-y|$, for all x,y, is called a contractive mapping. Using comparison with the geometric series of ratio s, the sequence of iterates $x_{n+1}=f(x_n)$ converges to a unique fixed point, defined by $f(x)=x$. This program can be used to find the root of an equation $g(x)=0$ provided that g(x) can be written as $g(x)=x-f(x)$ and $|f'(x)|<1$ in the region of interest.

Algorithm for Contraction Mapping

f, the function, is supplied at line 70.

x_0, the initial guess, is entered as requested.

The calculation is expressed as follows:

c=1,a(i)=0, i=0,1,2 initialize counter, iterates to prevent oscillation.

$x_1=f(x_0)$

c=c+1

test for stopping:	disp x_1,stop	if $\|x_1-x_0\|<$EPS
	a (mod (c,3))=x_1	
	disp $(x_1+x_0)/2$	if $\|a(2)-a(0)\|<$EPS and if $\|x_1-x_0\|\leq$EPS
	$x_0=x_1$	if $\|a(2)-a(0)\|\leq$EPS
	iterate again	

[EPS= 1E-499 is the smallest available number]

SAMPLE PROBLEM

A) Find the fixed point of $f(x)=1+\frac{1}{2}\arctan(x)$. Since $|f'(x)|<\frac{1}{2}$ for all x, f is contractive and so has a fixed point.

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
A 1	Enter f(x) in line 70	70 DEF FNF(X)=1+ATN(X)/2	
2	Run program	***CONTRACTION MAPPING***	
3	Enter initial guess	Initial guess?	1 [RTN]
		Current iterates roll	
4	Answer	Fixed point =1.48982393006	[RTN]
5	End	Run again, View again, or End? R	[RTN]

SAMPLE PROBLEM

B) Find the fixed point of $f(x)=\sqrt{x+2}$ for $x\geq 0$. Since $|f'(x)|<\frac{1}{2}$ for $x\geq 0$, f is contractive. Note that the fixed point satisfies the equation, $x^2-x-2=0$.

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
B_1	Enter f(x) in line 70	70 DEF FNF(X)=SQR(2+X)	
2	Run program	***CONTRACTION MAPPING***	
3	Enter initial guess	Initial guess?	1 [RTN]
		Current iterates roll	
4	Answer	Fixed point =2	[RTN]
5	End	Run again, View again, or End? R	E [RTN]

SAMPLE PROBLEM

C) Find the smallest positive root of EXP(-x)-SIN(x).
A fixed point of f(x)=x+EXP (-x)-SIN(x) will be a root.
The derivative, f'(x)=1-EXP(-x)-COS(x), is itself increasing, so on [a,π/2], in which a>0, |f'(x)|<1 and f is a contraction on [a,π/2]. Now find a couple of more roots.

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
C 1	Enter f(x) in line 70	70 DEF FNF(X)=X+EXP(-X)-SIN(X)	
2	Run program	***CONTRACTION MAPPING***	
3	Enter initial guess from		
	(a,π/2)	Initial guess?	1 [RTN]
4	Answer	Fixed point =.58853274398	[RTN]
5	Run again	Run again, View again, or End? R	[RTN]
6	Enter initial guess	Initial guess	4 [RTN]
7	Answer	Fixed point =6.28504927338	[RTN]
8	Run again	Run again, View again, or End?	[RTN]
9	Enter initial guess	Initial guess	10 [RTN]
10	Answer	Fixed point =12.5663741017	[RTN]
11	End	Run again, View again, or End?	E [RTN]

SAMPLE PROBLEM

D) Find the root of $x^4-4x^3+2x^2-4x+1=0$ on [0,1]. In fact, there is a root on [0,1] since the value of the polynomial at 0 and at 1 changes sign. Factoring x and placing -1 on the right hand side of the equation yields $x(x^3-4x^2+2x-4)=-1$. So we find the fixed point of $f(x)=-1/(x^3-4x^2+2x-4)$ on [0,1].

SOLUTION

STEP	INSTRUCTIONS	DISPLAY	INPUT
D 1	Enter f(x) in line 70	70 DEF FNF(X)=-1/(X^3-4*X^2+2*X-4)	
2	Run program	***CONTRACTION MAPPING***	[RTN]
3	Enter initial guess	Initial guess?	.5 [RTN]
4	Answer	Fixed point =.267949192431	[RTN]
5	Run again	Run again, View again, or End? R	E [RTN]

USER INSTRUCTIONS

STEP	INSTRUCTIONS	DISPLAY	INPUT
1	Enter f(x) as user defined function in line 70	70 DEF FNF(X)= user function	
2	Run program	***CONTRACTION MAPPING***	[RTN]
3	Enter initial guess	Initial guess?	X_0 [RTN]
4	Answer	Fixed point =	
5	Repeat options	Run again, View again, or End?	R [RTN]
5a	To run again, press [RTN]		or
5b	To view the answer again,		V [RTN]
	enter V [RTN]		or
5c	To end the program		E [RTN]
	enter E [RTN]		

VARIABLE NAMES

NAME	DESCRIPTION	NAME	DESCRIPTION
FNF(X)	Function with contractive property	X_0	Initial guess
A(I)	Mini array saves last 3 iterates, i=0,1,2	X_1	Iterate
C	Counter for setting A(I)	Q$	Control string for RVE module

NOTES AND REFERENCES

References: Braun, M., DIFFERENTIAL EQUATIONS AND THEIR APPLICATIONS, (Springer-Verlag, New York, 1978).

Conte, S.D., and deBoor, C., ELEMENTARY NUMERICAL ANALYSIS, (McGraw-Hill, New York, 1980).

deBoor, C. and Rosser, J.B., POCKET CALCULATOR SUPPLEMENT FOR CALCULUS, (Addison-Wesley, Reading, 1979).

PROGRAM LISTING

```
 10 ! CONTRACTION MAPPING
 20 ! Finds the fixed point of a
 30 ! contractive mapping
 40 DISP '* * * CONTRACTION MAPPING * *
     *'
 50 ! Revision 11/01/1982
 60 ! Define function
 70 DEF FNF(X) = X+EXP(-X)-SIN(X)          -User defined function
 80 ! Initialize counter and
 90 ! stopping check
100 C=1 @ A(0)=0 @ A(1)=0 @ A(2)=0         -Initialize counter, array for
                                            oscillatory iterates
110 ! Initial guess
120 INPUT 'Initial guess?';X0
130 ! Iterate and step counter
140 X1=FNF(X0)
150 C=C+1
160 DISP 'Current iterate=';X1
170 ! Stopping criterion and
180 ! check for oscillatory
190 ! behavior of approximation
200 IF ABS(X1-X0)<EPS THEN 230
210 A(MOD(C,3))=X1                         -Define A(1) -> A(3) to be the
                                            current iterate
220 IF ABS(A(2)-A(0))<EPS THEN 230 ELSE    -Check for oscillatory iterates
     X0=X1 @ GOTO 140
230 DISP 'Fixed point=';X1
240 ! Wait for return key to continue
250 IF NUM(KEY$)#13 THEN 250 ELSE 260
260 ! Run,view, or end                      -Continuation options
270 DISP CHR$(210);'un again,';CHR$(214
    );'iew again,or ';CHR$(197);
280 INPUT 'nd?','R'; Q$ @ Q$=UPRC$(Q$[1
    ,1])
290 ON POS('RVE',Q$)+1 GOTO 270,120,230
    ,300
300 STOP
```

NOTES

MATH II

MIDPOINT RULE FOR INTEGRATION
TRAPEZOIDAL RULE FOR INTEGRATION
ROMBERG RULE FOR INTEGRATION
SIMPSON'S RULE FOR INTEGRATION
NEWTON—COTES RULE FOR INTEGRATION
EULER'S METHOD
NEWTON'S METHOD
TRAPEZOIDAL RULE FOR ORDINARY DIFFERENTIAL EQUATIONS
RUNGE-KUTTA
CONTRACTION MAPPING

ALL HP-75 SOLUTIONS BOOKS ARE AVAILABLE RECORDED ON MINI-DATA CASSETTES FROM EITHER A HEWLETT-PACKARD DEALER OR THE HP USERS' LIBRARY.

00075-90062 Printed in U.S.A.

Reorder No.
00075-13004