Problem Set 2

```
> restart:
```

Exact Solution to ODE/BVP

```
> X := t -> (3/2)*t + exp(2*t)/4 + 3/4;
```

$$X := t \to \frac{3}{2} t + \frac{1}{4} \mathbf{e}^{(2\,t)} + \frac{3}{4}$$

Exact value at t = 0.4 (to 13 decimal places)

```
> ev := evalf(X(0.4),13);
```

$$ev := 1.906385232123$$

```
> #ev := evalf(X(4),13);
>
>
```

Problem 1. Euler Method

```
>
```

Initializations:

```
> F := (t,x) -> 2*x - 3*t;   # function on right hand side of Diff
  E.
```

$$F := (t, x) \to 2\,x - 3\,t$$

```
> t[0] := 0.0:  # initial value of t
> x[0] := 1.0:  # initial value of x
> h := 0.1: # step size
> tf := 0.4:   # final value of t
>  N := floor((tf - t[0])/h);  # number of steps
```

$$N := 4$$

The loop:

```
> for i from 1 to 4 do
>   t[i] := t[i-1] + h:
>   x[i] := x[i-1] + h*F(t[i-1],x[i-1]):    # the Euler method
  formula
>   print(t[i],x[i]):
> od:
```

0.1, 1.20

0.2, 1.410

0.3, 1.6320

0.4, 1.86840

The result

```
> t[N]; x[N]; print("Error = ", ev - x[N]);
```

0.4

1.86840

"Error = ", 0.037985232

>

(b) Repeat using step size of 0.01.

```
> dt := 0.01;
  N := floor((tf - t[0])/dt);
```

$$dt := 0.01$$

$$N := 40$$

The loop:

```
> for i from 1 to N do
>   t[i] := t[i-1] + dt:
>   x[i] := x[i-1] + F(t[i-1],x[i-1])*dt:
>   if type(i/10,`integer`) then print(t[i],x[i]): fi:
> od:
```

0.10, 1.204748605

0.20, 1.421486850

0.30, 1.652840396

0.40, 1.902009916

The result:

```
> t[N]; x[N]; print(`Error = ` , ev - x[N]);
```

0.40

1.902009916

Error = , 0.004375316

>

>

>

>

Problem 2: Huen Method (Second Order Runge-Kutta)

>

Initialization:

```
> F := (t,x) -> 2*x - 3*t:   # function on right hand side of Diff
  E.
> t[0] := 0:  # initial value of t
> x[0] := 1:  # initial value of x
> h := 0.01: # step size
> tf := 0.4:   # final value of t
>  N := floor((tf - t[0])/h);  # number of steps
```

$$N := 40$$

The loop:

```
> for i from 1 to N do
```

```
>   t[i] := t[i-1] + h:
>   f1 := F(t[i-1],x[i-1]):
>   f2 := F(t[i-1] + h, x[i-1] + h*f1):
>   x[i] := x[i-1] + (h/2)*(f1 + f2):
>   if type(i/10,`integer`) then print(t[i],x[i]): fi:
> od:
```

0.10, 1.205346678

0.20, 1.422946377

0.30, 1.655511751

0.40, 1.906356001

```
> t[N]; x[N]; print(`Error = `, ev - x[N]);
```

0.40

1.906356001

Error = , 0.000029231

```
>
>
>
```

Problem 3: Fourth Order Runge-Kutta

Initialization:

```
> F := (t,x) -> 2*x - 3*t:   # function on right hand side of Diff
  E.
> t[0] := 0:  # initial value of t
> x[0] := 1:  # initial value of x
> h := 0.01: # step size
> tf := 0.4:   # final value of t
>  N := floor((tf - t[0])/h);  # number of steps
```

$N := 40$

The loop:

```
> for i from 1 to N do
>   t[i] := t[i-1] + h:
>   f1 := F(t[i-1],x[i-1]):
>   f2 := F(t[i-1] + h/2, x[i-1] + (h/2)*f1):
>   f3 := F(t[i-1] + h/2, x[i-1] + (h/2)*f2):
>   f4 := F(t[i-1] + h, x[i-1] + h*f3):
>   x[i] := x[i-1] + (h/6)*(f1 + 2*f2 + 2*f3 + f4):
>   if type(i/10,`integer`) then print(t[i],x[i]): fi:
> od:
> t[N];
> x[N];
> print(`Error = `,ev - x[N]):
>
```

0.10, 1.205350690

0.20, 1.422956175

0.30, 1.655529702

0.40, 1.906385233

0.40

1.906385233

Error = , -0.1 10^{-8}

```
>
>
```

Problem 4: Adams-Bashforth Method

For this method we need four initial data points. We will grab the first four data points of the preceding Runge-Kutta calculation (Problem 3) and just run the loop starting at i=4

The loop:

```
> for i from 4 to N do
>    t[i] := t[i-1] + h:
>    x[i] := x[i-1] + (h/24)*( 55*F(t[i-1],x[i-1]) -
  59*F(t[i-2],x[i-2]) + 37*F(t[i-3],x[i-3]) - 9*F(t[i-4],x[i-4])  ):
>   if type(i/10,`integer`) then print(t[i],x[i]): fi:
> od:
>
> t[N];
> x[N];
> print(`Error = `,ev - x[N]):
```

0.10, 1.205350687

0.20, 1.422956168

0.30, 1.655529687

0.40, 1.906385212

0.40

1.906385212

Error = , 0.20 10^{-7}

```
>
>
>
>
>
```