# COMPUTERIZED NUMERICAL SOLUTION OF ORDINARY DIFFERENTIAL EQUATION

BY

ONUORAH MARTINS ONYEKWELU
PGD/MSC/626/97/98

DEPARTMENT OF MATHEMATICS/COMPUTER SCIENCE
FEDERAL UNIVERSITY OF TECHNOLOGY, MINNA

IN PARTIAL FULFILLMENT OF THE REQUIREMENT
FOR THE AWARD OF POST-GRADUATE DIPLOMA (PGD)
IN COMPUTER SCIENCE

NOVEMBER 2003

# CERTIFICATION

This is to certify that this research project was carried out by

ONUORAH MARTINS ONYEKWELU
PGD/MSC/626/97/98

| | |
|---|---|
| .......................................... | .................................. |
| PROFESSOR K. R. ADEBOYE | MR. L. N. EZEAKOR |
| PROJECT SUPERVISOR | HEAD OF DEPARTMENT |

# DEDICATION

This work is dedicated to my parents Mr. Albert Udekwe Onuorah and my siblings.

# ACKNOWLEDGEMNT

I am most grateful to God Almighty for the life he gave to me, his mercy, protection, provision and guidance throughout the duration of my programme.

I also wish to acknowledge my supervisor Professor K. R. Adeboye for his efforts by way of supervision and necessary corrections that resulted to this piece of work. To other lecturers in the department, I remain grateful for their respective inputs.

I wish to appreciate the entire Onuorah Anijah's family for their financial support, love for me and for one another.

I thank God Almighty for the persons of Mr. and Mrs. J. O. Ikebudu, Mr. Nnamdi Okafor, Mr. Anthony Eze, Mr. Daniel Haruna and many others that space will not permit me to enlist here, I wish them success in all their endeavours

# TABLE OF CONTENTS

## CHAPTER ONE

INTRODUCTION

1.1 HISTORICAL BACKGROUND OF DIFFERENTIAL EQUATION...... 1

1.2 AIMS AND OBJECTIVES OF THE PROJECT..............................3

1.3 SCOPE AND LIMITATION OF THE PROJECT............................4

1.4 OUTLINE OF THE PROJECT...................................................4

## CHAPTER TWO

DEFINITION OF TERMS AND SOME ANALYTICAL SOLUTIONS

2.1 DIFFERENTIAL EQUATIONS................................................6

2.2 DEFINITON OF TERMS.......................................................6

2.3 FIRST ORDER ORDINARY DIFFERENTIAL EQUATION.............8

2.4 ANALYTICAL SOLUTION OF ORDINATY DIFFERENTIAL EQUATION ......................................................................8

## CHAPTER THREE

ALGORITHM DEVELOPMENT

3.1 ALGORITHM FOR EULER'S METHOD..................................14

3.2 ALGORITHM FOR TAYLOR'S SERIES METHOD.......................16

3.3 ALGORITHM FOR HEUN'S METHOD...................................18

3.4 ALGORITHM FOR RANGE-KUTTA METHOD........................20

## CHAPTER FOUR

MPLEMENTATION

.1 SOFTWARE AND HARDWARE RQUIREMENT.........................21

.2 TYP AND FEATURS OF THE PROGRAM...............................21

.3 MENU DESCRIPTION..........................................................21

## CHAPTEER FIVE

## CONCLUSION

5.1 OUTPUT OF PROGRAM........................................................23

5.2 ANALYSIS OF THE METHODS..............................................23

5.3 RECOMMENDATION..........................................................24

APPENIX A - PROGRAMMING

# CHAPTER ONE

# INTRODUCTION

## 1.1 HISTORICAL BACKGROUND:

Without knowing something about differential equations and methods of solving them, it is difficult to appreciate the history and development of this important branch of mathematics. Further, the development of differential equations is intimately interwoven with the general development of mathematics and cannot be divorced from it.

In this brief comment, I mentioned only the most prominent of the mathematicians of the seventeenth and eighteenth Centuries who made important contributions to the theory; methods of solution and application of differential equations.

The study of differential equations originated in the beginning of the calculus with Isaac Newton (1642 – 1727) and Gottfried Wilhelm Leibniz (1646 – 1716) in the seventeenth Century. Newton educated at Trinity College, Cambridge and later become a Professor there did relatively little work in differential equation, but his development of calculus and elucidation of the basic principles of mechanics provided a basis for their application in the eighteenth century by Euler. Newton classified first order elementary differential equation according to the form dy/dx = f(x) and dy/dx = f(x,y). For the later equation, he developed a method of solution using infinite series when f(x,y) is a polynomial in x and y.

Leibniz, born at Leipzy, arrived at the fundamental results of calculus independently, although a little later than Newton, but was first to publish them in 1684.

Leibniz was very conscious of the power of good mathematical notation and our notation for derivatives dx/dy and the integral sign $\int$ are due to him. He discovered the method of separation of variable, the reduction of homogenous equations to separable ones, and the procedure for solving first order linear equations.

The brothers Jakob (1654-1705) and Jonhann (1667 – 1748) Bernoulli of Basel did much to develop methods of solving differential equations and to extend the range of application. For example, Jakob Bernoulli solved the differential equation $y^1=[a^3/(b^2– ya^3)]^{½}$ in 1690 and in the same paper first used the term "integral" in the modern sense. In 1694 Johanna was able to solve the equation $dy/dx = y/ax$ even though it was not yet known that $d(\ln x) = dx/x$. the two brothers collectively solve the Brachisetochrone problem (the determination of the curve of fastest descent) which heads to the nonlinear first order equation $y\ [1+(y^1)^2] = c$.

Daniel Bernoulli (1700 – 1782), son of Johann educated at St. Petersburg Academy, become a professor of botany 1733 and later of physics. His interests were primarily in partial differential equations and their applications. For instance, it is his name that is associated with the famous Bernoulli equation in fluid mechanics. He was first to encounter the function that a century later becomes known as Bassel functions.

The greatest mathematician of the eighteenth Century, Leon hard Euler (1707 – 1783), was a student of Johann Bernoulli, associated with St. Petersburg Academy (1777- 1783) and Berlin Academy (1741 – 1766). Euler was the first the most prolific mathematician of all time. His interests ranged over all areas of mathematics and many fields of applications. Of particular interest, here is his formation of problems in mechanics in mathematical languages and his development of methods of solving these mathematical problems. Among other things, Euler identified the condition for exactness of first order differential equations, developed the theory of integrating factor in the same paper, and gave the general solution of homogenous linear equation with constant coefficient. He also proposed a numerical procedure (chapter 3 section 3.1) in 1768 – 69, made important treatment of the calculus variation.

Joseph Louis Lagrange (1736 – 1813) becomes a professor of mathematics at the age of 19. With respect to elementary differential equations, Lagrange showed in 1762 – 65 that the general solutions of an nth order linear homogenous differential equation is a linear combination of an n independent solutions, he later on in 1774 – 75 gave complete development of the method of variation of parameter.

Pierre Simon de Laplace (1749 – 1827), he uses mathematics as a tool for understanding nature. His works among others is the Laplace transform, named after him, which is used to solve differential equations.

## 1.2 AIMS AND OBJECTIVES OF THE PROJECT

It is obvious that some of the differential equations occurring in nature such as;

a. The growth rate per individual in a population.

b. A bacterial population that is changing at a rate proportional to the size.

c. Newton's law of cooling, which states that the rate of change of temperature difference between an object and its surrounding medium is proportional to the temperature difference, and many others do not admit of analytical solution.

The method amenable to the solution of such differential equations is the numerical method. This numerical method is an approximation method, and when used manually, is error prone, labour intensive, time-consuming etc. The advent of computers has led to an increased emphasis on numerical techniques for finding very accurate approximate solutions, to these problems.

The aim of this project work is to therefore, develop a computer-based program for all the numerical methods stated which would solve the associated problems with high level of accuracy, speed, efficiency etc. It is also intended to make comparisms of the various methods used and make recommendations.

## 1.3 SCOPE AND LIMITATION

Ordinary differential equations can be classified as first order, second order and higher order.

$y^1 = 1 + x + y$ is an ordinary differential equation of first order which can be written in function form as $f(x, y, y^1) = x + y + 1 - y^1 = 0$.

$2y^1 + 3xy^1 + y = e^x$ is an ordinary differential equation of second order which can be also written in function form as $f(x, y, y^1) = e^x - y - 3xy - 2y^1 = 0$ and so on.

For the purpose of this work, I limited myself on the numerical solution of first order ordinary differential equation using the stated methods. In order words, the numerical solution of second and higher order equations and other methods not discussed are above the scope of this project.

## 1.4 OUTLINE OF THE PROJECT

Chapter one basically introductory in nature, it covers the historical development of differential equations within the seventeenth and eighteenth centuries. The aim and objectives of the project work as well as the scope and limitation of the project were also stated. In chapter two, I introduce differential equations, definition of terms and classification of differential equations, which is the bone of contention. There was also an attempt to present some analytical solution of first order ordinary differential equations.

Chapter three, contains the algorithm (the procedure for solving a problem in a finite number of steps) for all the methods discussed. These Algorithms were represented using a flow chart. The chapter is a combination of the first and second stages of program development (program planning and program design).

On the other hand, chapter four contains the translation of the design made in chapter three into the choice of programming language (Visual Basic) and its application to the main problem. It also contains the software/hardware specification, the nature and type of the program.

Finally, chapter five contains the output of the program in chapter four, the analysis of the methods used and references.

# CHAPTER TWO

# DEFINITION OF TERMS AND SOME ANALYTIC SOLUTIONS

## 2.1 DIFFERENTIAL EQUATIONS

Many basic laws of chemistry, physics engineering, medicine and even social sciences are defined in mathematical terms that contain unknown functions and their derivatives. Any equation relating an unknown function and its derivative is called a differential equation. Frequently mathematics are needed for solving problems that include variable that has continuous pattern of variation (e.g. time and motion), and usually differential equations serve as the model. The next section of this chapter introduces some basic definition and concepts regarding elementary differential equations in general while section 2.3 and the rest of the work were devoted to first order ordinary differential equations and their numerical solutions.

## 2.2 **DEFINITION OF TERMS**

**Definition 2.2.1:** Ordinary differential equation is an equation that contains an independent variable $x$, and contain derivatives of $y$ such as $y^1(x)$, $y^{11}(x)$, .......... $y^{(n)}(x)$.

**Definition 2.2.2:** A function expression that is a combination of the variable $x$, the unknown function $y(x)$, and the derivatives as $f(x, y^1\ y^{11}, y^{111}......y^{(n)}) = 0$. when written in this form, it is called an ordinary differential equation of order $n$, where $n$ is a positive integer.

**Definition 2.2.3:** The highest order of a derivative (of the unknown function) that appear in a differential equation is also the order of that equation the following examples illustrates differential equation of order 1, 2 and 3 $y^1 = 1 + x + y$ is and ordinary differential equation of order one, it can be written in function form as $f(x, y, y^1) = x + y + 1 - y^1 = 0$.

$2y^{11} + 3xy^{1} + y = e^{x}$ which is an ordinary differential equation of order two and can be written in function form as $f(x, y^{1}, y^{11}) = e^{x} - y - 3xy^{1} - 2y^{11} = 0$ and finally $xy^{111} - 2y^{1} + e^{x}y = /n/x/$ is a third order ordinary differential equation.

**Definition 2.2.4:** For a differential equation order one, a solution is a function f (x) defined on an open intervals I such that $f(x, y^{1}(x)\ldots y^{(n)}(x) = 0$ for every $x$ in the interval I.

**Definition 2.2.5:** An equation of order n is said to be linear if it has the special form

$a_o(x)\, y^{(n)} + a_1(x)\, y^{(n-1)} + \ldots\ldots\ldots\ldots + a_{n-1}(x)\, y^{1} + a_n(x)\, y = f(x)$ where $a_i$ are arbitrary functions of $x$ only. Also note that in this form the unknown function y and all of its derivatives appear linearly, (i.e. none may have a power higher than one and none may be multiplied by any of the others). The coefficients $a_o(x), a_1(x), \ldots\ldots\ldots\ldots a_n(x)$, and $f(x)$ must not contain $y$ or any of its derivatives. The equation is non – linear if it cannot be put into the form above.

**Definition 2.2.6:** If all the coefficient functions are constant functions, $a_o(x), a_1(x), \ldots\ldots\ldots\ldots a_n(x)$are constant functions as in $3y^{11} + 6y^{1} + y = e^{x}$, the equation is said to have constant coefficient. If any one of the coefficient function is not a constant function, then the equation is said to have variable coefficients.

**Definition 2.2.7:** A linear equation is called homogenous if $f(x) \doteq 0$ and non homogenous if $f(x) \neq 0$. a non-linear equation is homogenous if it contains no terms that are function of $x$ alone otherwise, it is non-homogenous.

**Definition 2.2.8:** A solution of an ordinary differential equation of order $n$ is a function that possesses at least $n$ derivatives on some interval I and that satisfies the equation.

**Definition 2.2.9:** An initial value problem is a differential equation in which the values of the function and some of its derivatives at one point, called the initial point, are specified.

**Definition 2.2.10:** System of differential equation is a differential equation with more than one variable.

## 2.3 FIRST ORDER DIFFERENTIAL EQUATION

A differential equation involving only ordinary derivatives (derivatives of function of one variable) is called ordinary differential equation.

This ordinary differential equation can be of order 1, 2 ............ , $n$ which is the highest order of a derivative (of the unknown function) that appears in a differential equation. A first order ordinary differential equation is one like $y^1 = 1 + x + y$ which can be written in function form as $f(x, y, y^1) = x + y + 1 - y^1 = 0$. As stated earlier, this will form the subject of this study work.

## 2.4 ANALYTIC SOLUTION OF ORDINARY DIFFERENTIAL EQUATION

In this section we will look at some analytic method of solving first order ordinary differential equation of the form $y^1 = f(x, y)$. i.e. we find a unique function $y(x)$ satisfying $y^1(x) = f(x, y)$ and $y(x_o) = y_o$.

Solving a first order ordinary differential equations of the form $y^1 = f(x, y)$ can be very difficult at times since there is no general method that can be used for all cases. In this section, we base our discussion on the most useful methods.

### 2.4.1 DIRECT METHOD

Consider the first ordinary differential equation $y^1 = f(x, y)$.

If the function does not involve the variable $y$, then the equation can be solved by integrating both sides with respect to $x$.

Example 1

Solve $y^1 = x$.

Integrate both sides of the equation with respect to $x$, we have

$$\int y^1 dx = \int x dx$$

Changing the variable of integration on left hand side, we obtain

$$y = \int dy = \int x dx = \frac{x^2}{2} + c$$

Example 2

Solve the equation $y^1 = 2xy$

We rewrite this equation as

$$\frac{1}{y} dy = 2xdx$$

So that we may automatically perform the change of variable. Integrating both sides yields.

$$\text{In } y = x^2 + c$$

And exponentiation both sides of the equation, we have

$$y = \mathrm{e}x^2 + \mathrm{c} = \mathrm{e}^{\mathrm{c}}\, \mathrm{e}^{x2} = \mathrm{c}_1 \mathrm{e}^{x2}.$$

This is the general solution of the equation, and involves, an unspecified real constant $c^1$. Thus the differential equation $y^1 = 2xy$ has an infinite number of solution, depending on the particular choice of $c_1$.

### 2.4.2 SUBSTITUTION TECHNIQUES

In this section, we present some special substitutions that may be used occasionally to solve differential equations. Suppose that we are given a first order differential equation of the form

$$y^1 = f({}^{y}/_{x})\ldots\ldots\ldots(i)$$

That is the right – hand side of the equation can be written as a function of the variable y/*x*. It is then natural to try the substitution z = y/*x*. Since the function y depends on x, so does the function z.

Differentiating y = *x*z with respect to x. we have $y^1 = z + x\frac{dz}{dx}$……..(2).

Substituting $z = {}^{y}/_{x}$ and replacing the left-hand side of equation (2), we obtain

$$z + x\frac{dz}{dx} = F(z)$$

The variable in this equation can be separated, since $x\frac{dz}{dx} = F(z) - z$

So that $\frac{dz}{F(z) - z} = \frac{dx}{x}$

A complete solution can now be obtained by integrating both sides of this equation and replacing each z by y/*x*.

Example 1

Solve the equation $y^1 = \frac{x - y}{x + y}$

Dividing the numerator and denominator of right-hand side of the equation by *x*

were $y1 = \frac{1 - (y/x)}{1 + (y/x)} = \frac{1 - z}{1 + z} = F(z)$

Replacing the left-hand side by z + $x$(dz/d$x$) and separating variables, we have

$$\left(\frac{1 + z}{1 - 2z - z^2}\right) \quad dz = \underline{d^{x}x}$$

After integration, we have

$$\text{In}\ (1 - 2z - z^{Z} \qquad = -2\ \ln x + c = \text{In}\ (cx^{-2}),$$

Exponentiation and replacing z by y/$x$ we here the implicit solution

$$1 - \frac{2y}{x} - \frac{y^2}{x^2} = \frac{c}{x^{2.}}$$

Finally, multiplying both sides by $x^2$

$$x^2 - 2xy - y^2 = c.$$

# CHAPTER THREE

## ALGORITHM DEVELOPMENT FOR NUMERICAL SOLUTION OF FIRST ORDER ORDINARY DIFFERENTIAL EQUATIONS

Given an initial value problem (I. V. P.)

$$y^1 = (x, y). \ldots\ldots\ldots\ldots\ldots (3.1).$$

Over the interval [a, b], y(a) = y(0).

In actuality, we will not find a differentiable function y($x$) that will satisfy the (I.V.P.) instead, a set of points ($x_k$, $y_k$) are generated which are used for an approximation (i.e $y(x_k) = y_k$) for convenience, we divide the interval [a, b] into m equal intervals and select the mesh points.

$$x_k = a + hk \text{ for } k = 0,1,2,\ldots\ldots\ldots,m,$$

$$\text{Where } h = \frac{b - a}{m}$$

The intent of this chapter therefore is to present an ALGORITHM for all the methods under discussion which when translated into visual basic programming language, will obtain a very accurate solution of the associated I. V. P.

An ALGORITHM is a procedure for solving a problem in a finite number of steps. It can also be defined as a finite set (or sequence) for carrying out a specific task or solving a problem. On the other hand, a program is an ALGORITHM specifically expressed in high-level language capable of execution by a computer.

In general, an ALGORITHM can be expressed in any of the following terms.

a. Pseudcodes (Structured English)

b. Flow Chart

c. Nassi – Schnedman structured flow (NSSF) diagram

d. English language

e. Decision tree

f. Formulae

g. Warmer – Orr diagrams.

For the purpose of this study work, flow charts will be used in representing the ALGORITHM.

A flow chart is composed of symbols that represent specific activities.

## 4.1 EULER'S METHOD

The general step for using Euler's method to solve first order ordinary differential equation of the form $y^1 = f(x, y)$ over the interval [a, b] given that y(a) = y(0) is

$y_{k+1} = y_k + hf(x_k, y_k)$, $x_{k+1} = x_k + h$

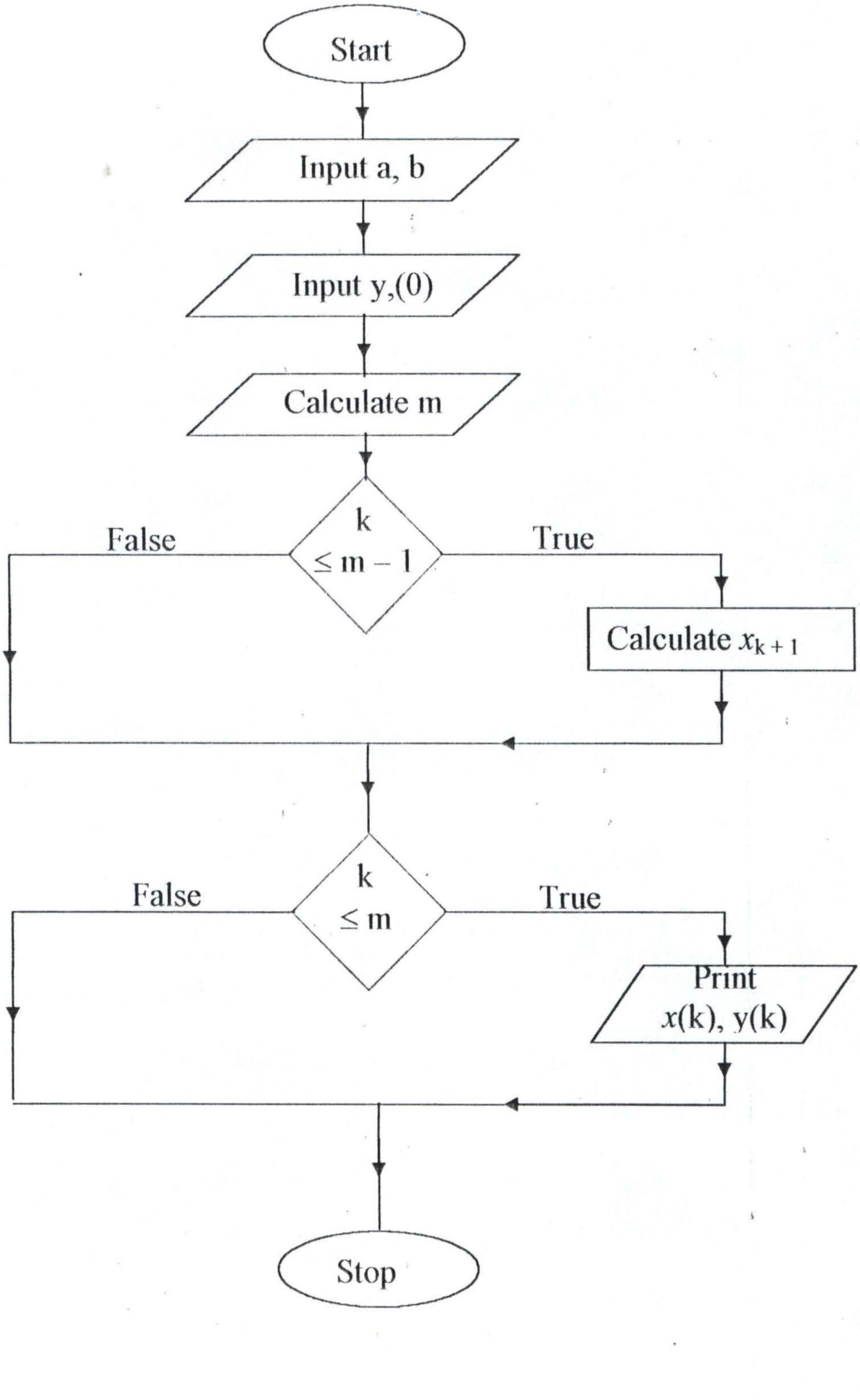
For k = 0, 1, 2, ............., m – 1.

Example 4.1

Use Euler's method to solve approximately the I. V. P $y1 = \frac{x-y}{2}$ on [0, 3]

With y(0) = 1, h = 0.25.

FLOW CHART FOR EULER'S METHOD



## 4.2 TAYLOR'S SERIES METHOD

The Taylor's series method is of general applicability and it gives a means to compare the accuracy of the various methods for solving I. V. P. It can be constructed to have a higher degree of accuracy.

The general step for Taylor's series method of order N is

$$y_{k+1} = y_k + d_1 h + \frac{d_2 h^2}{2!} + \frac{d_3 h^3}{3!} + \ldots\ldots\ldots + \frac{d_N h^N}{N!}$$

Where $d_j = y^{(j)}(x_k)$ for $j = 1, 2, \ldots\ldots\ldots\ldots\ldots, N$,

$K = 0, 1, \ldots\ldots\ldots\ldots\ldots, m - 1$

Example 4.2

Use Taylor's series method of order N = 4 to solve the I. V. P. $y^1 = \frac{x - y}{2}$ on [0,3],

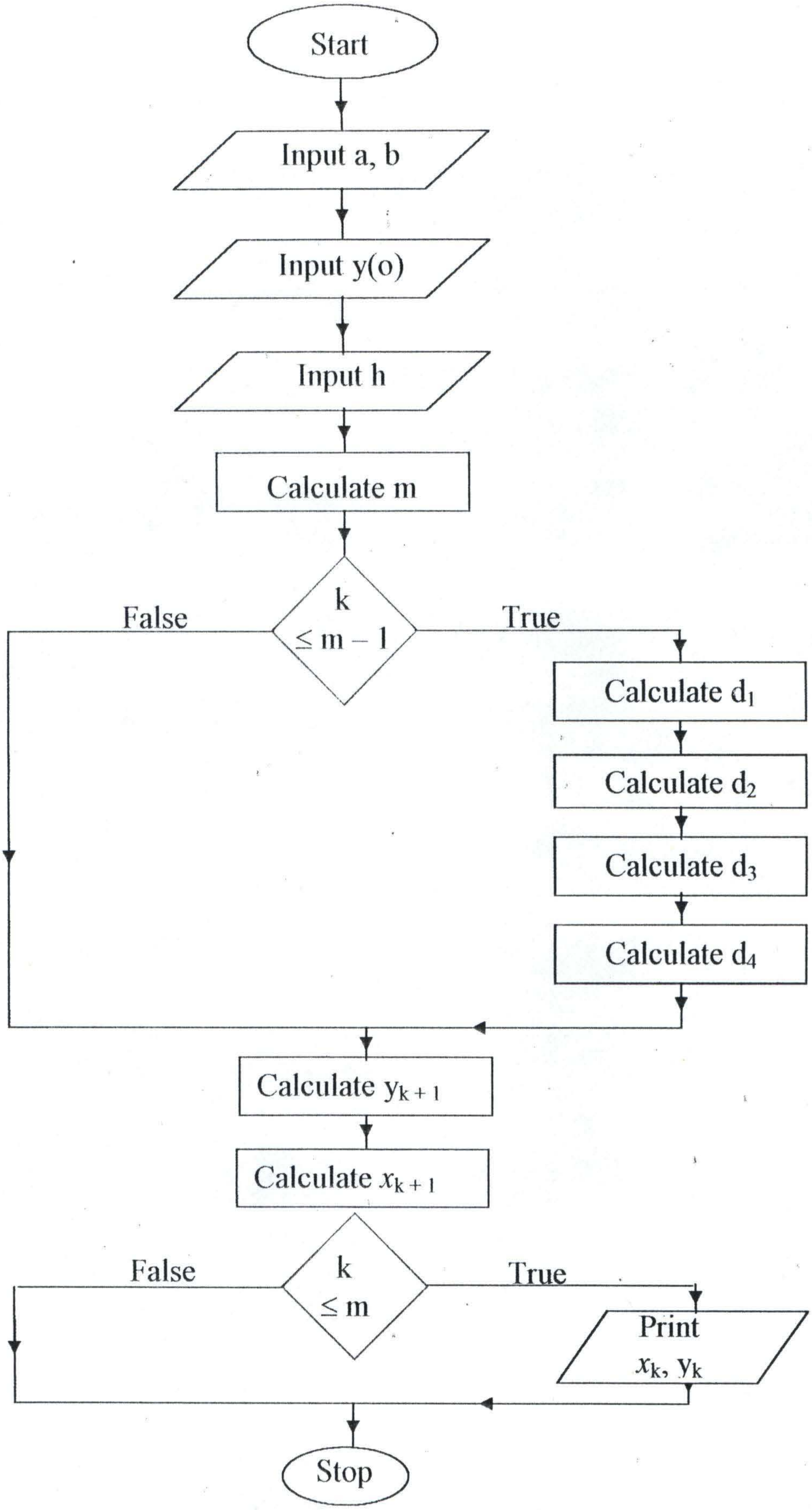
With $y(0) = 1$, $h = 0.25$

Solution

The derivatives of $y(x)$ must first be determined. Recall that the solution $y(x)$ is a function of $x$, and differentiate the formula $y^1(x) = f(x, y(x))$ with respect to $x$ to get $y^2(x)$.

Then continue the process to obtain the higher derivatives.

## FLOW CHART FOR TAYLOR'S METHOD



## 4.3 HEUN'S METHOD

Heun's introduces a new idea for constructing an algorithm to solve the I. V. P. of the form $y^1 = f(x, y)$ over the interval

[a, b], $y(a) = y(0)$.

The general step for Heun's method is

$$y_{k+1} = y_k + \frac{h}{2}\left[f(x_k, y_k) + f(x_{k+1}, p_{k+1})\right]$$

Where $p_{k+1} = y_k + hf(x_k, y_k)$,

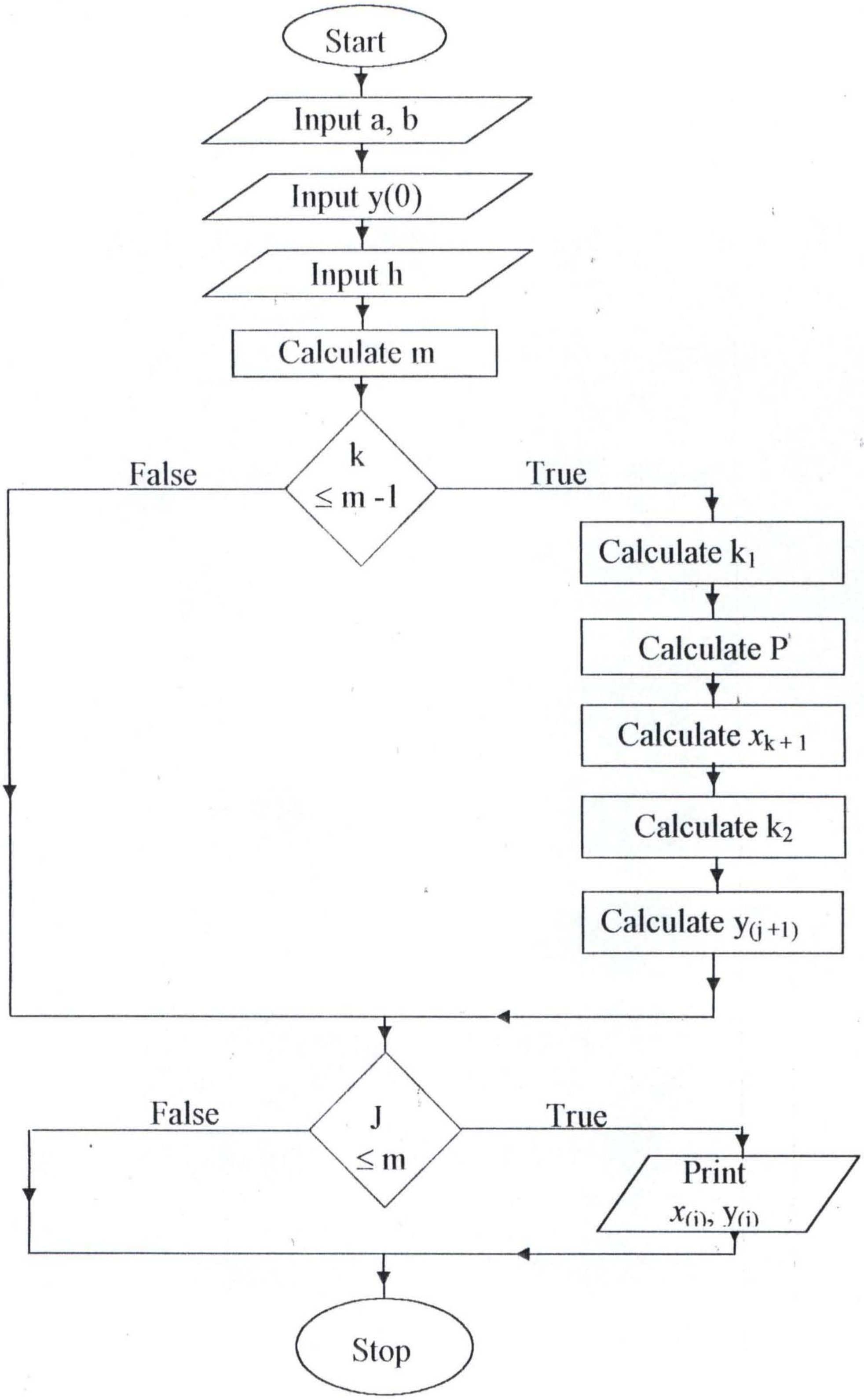
$$x_{k+1} = x_k + h$$

Example 4.3

Use Heun's method to solve the I. V. P. $y1 = \frac{x-y}{2}$

$h = 0.25$

FLOW CHART FOR HEUN'S METHOD



## 4.4 RUNGE – KUTTA METHOD

The Runge – Kutta method of order 4 simulates the accuracy of the Taylor series method of order 4. As before, the interval [a, b] is divided into m subintervals of equal width h starting with $(x_o, y_o)$, four function evaluations per step are required to generate the discrete approximation $(x_k, y_k)$ as follows

$$y_{k+1} = y_{k+h} (f_1 + 2\ f_2 + 2f_3 + f_4)$$

where

$$f_1 = (x_k, y_k),$$

$$f_2 = f(x_k + \frac{h}{2}, y_k + \frac{h}{2} f_1),$$

$$f_3 = f(x_k + \frac{h}{2}, y_k + \frac{h}{2} f_2)$$

$$f_4 = f(x_k + h, y_k + hf_3).$$

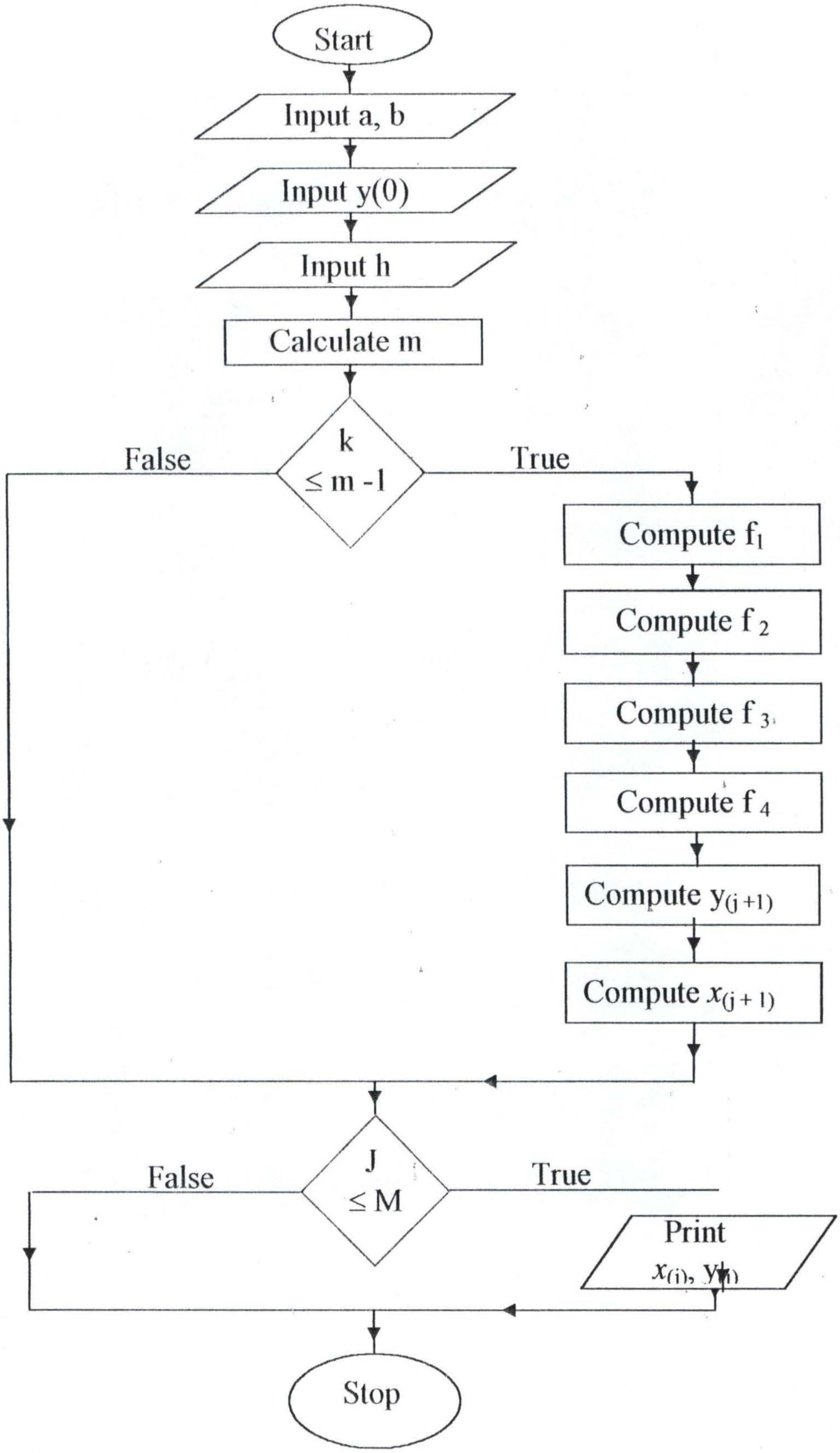
Example 4.4

Use Runge – Kutta method of order 4 to solve the I. V. P.

$$y^1 = x - y \text{ on } [0, 3,] \text{ with } y(0) = 1$$

$$h = 0.25$$

## SOLUTION
## FLOW CHART FOR RUNGE – KUTTA METHOD



# CHAPTER FOUR

## IMPLEMENTATION

### 4.1 SOFTWARE AND HARDWARE REQUIREMENT

The software requirement of this program is the task of transferring the design made in the previous chapter into visual Basic programming language. The program on the other hand requires a complete computer system and a disk space of 2.016MB.

### 4.2 TYPE AND FEATURES OF THE PROGRAM

The program is written in visual Basic, which is the latest version of Basic programming language. It is an object-oriented programming language (OOPL) and thus, event driven. It has more additional facilities than Q basic and the earlier version of basic.

The program in its finished form is a standard executable program, which can be run on windows operating system. Before the program can be run on any computer, it has to be set up as an executable program in that computer.

### 2.3 MENU DESCRIPTION

From the main screen, if the program is double clicked, or selected from program, the first thing that will appear is the welcome screen.

WELCOME TO. . .

NUMERICAL SOLUTION OF ORDINARY DIFFERENTIAL EQUATIONS

BY

ONUORAH MARTINS O.

REG. NO. PGD/MCS/626/97/98

Click here<br>
To continue

If you respond by clicking at the specified point, you will open to the main menu thus:

EULER'S METHOD

TAYLOR'S SERIES METHOD

RUNGE – KUTTA METHOD

HEUN'S METHOD

EXIT

A click on any of the above methods will take you to the object view of that particular method chosed, where you can key in your input. The object view of all the methods are the same and will have the format below:

Interval (□, □)

| k | $x_k$ | $y_k$ |
|---|---|---|

$y_{(o)}$ □

h = □

[Solve] [Clear entries] [clear table] [print] [close] [Exit]

After inputting the intervals, $y_{(o)}$ and h, a click on solve will present the result of the particular method on the table and will arrange the value of k, $x_k$ and $y_k$ appropriately as shown above. A click on clear entries will clear the values inputted, a click on clear table will equally clear the values of k, $x_k$ and $y_k$, a click on print will print the result in the table.

# Numerical Solutions of Differential Equations

## (Heun's Method)

=0.25

| k | Xk | Yk |
|---|---|---|
| *0* | *0* | *1* |
| 1 | 0.25 | 0.8984375 |
| 2 | 0.5 | 0.83807373046875 |
| 3 | 0.75 | 0.814080715179443 |
| 4 | 1 | 0.822196256369352 |
| 5 | 1.25 | 0.85865762270987 |
| 6 | 1.5 | 0.920143036171794 |
| 7 | 1.75 | 1.00371999945492 |
| 8 | 2 | 1.10679972730577 |
| 9 | 2.25 | 1.22709662932903 |
| 10 | 2.5 | 1.36259316001087 |
| 11 | 2.75 | 1.51150803640485 |
| 12 | 3 | 1.67226877063513 |

# Numerical Solutions of Differential Equations

## (Runge-Kutta Method)

h=0.25

| k | Xk | Yk |
|---|---|---|
| *0* | *0* | *1* |
| 1 | 0.25 | 0.897491455078125 |
| 2 | 0.5 | 0.836403667926788 |
| 3 | 0.75 | 0.811869581540425 |
| 4 | 1 | 0.819594034304221 |
| 5 | 1.25 | 0.855786550790071 |
| 6 | 1.5 | 0.917102057486772 |
| 7 | 1.75 | 1.00058853502075 |
| 8 | 2 | 1.10364082331459 |
| 9 | 2.25 | 1.22395987187823 |
| 10 | 2.5 | 1.35951680814227 |
| 11 | 2.75 | 1.50852112844587 |
| 12 | 3 | 1.66939274345836 |

# Numerical Solutions of Differential Equations

## (Taylor's Method)

=0.25

| k | Xk | Yk |
|---|---|---|
| *0* | *0* | *1* |
| 1 | 0.25 | 0.897491455078125 |
| 2 | 0.5 | 0.836403668237229 |
| 3 | 0.75 | 0.811869582013363 |
| 4 | 1 | 0.819594036125636 |
| 5 | 1.25 | 0.855786555353309 |
| 6 | 1.5 | 0.917102060548737 |
| 7 | 1.75 | 1.00058853508744 |
| 8 | 2 | 1.10364082001615 |
| 9 | 2.25 | 1.22395988312443 |
| 10 | 2.5 | 1.35951681805939 |
| 11 | 2.75 | 1.50852113895234 |
| 12 | 3 | 1.66939275154921 |

# Numerical Solutions of Differential Equations

## (Euler's Method)

25

| k | Xk | Yk |
|---|---|---|
| | *0* | *1* |
| | 0.25 | 0.875 |
| | 0.5 | 0.796875 |
| | 0.75 | 0.759765625 |
| | 1 | 0.758544921875 |
| | 1.25 | 0.788726806640625 |
| | 1.5 | 0.846385955810547 |
| | 1.75 | 0.928087711334229 |
| | 2 | 1.03082674741745 |
| | 2.25 | 1.15197339653969 |
| | 2.5 | 1.28922672569752 |
| | 2.75 | 1.44057337939739 |
| | 3 | 1.60425171256065 |

# CHAPTER FIVE

## CONCLUSION

### 5.1 OUTPUT OF THE PROGRAM

The output of the program can be easily generated since there is à code in the program that will enable the computer to print it.

The output of the specimen question $y^1 = \frac{x - y}{2}$ on [0,3] with $y_{(o)} = 1$, $h = 0.25$ is as shown in appendix B

### 5.2 ANALYSIS OF THE ALGORITHMS

For us to be able to make proper analysis of the methods used, it will be better if we make a table of the Global Truncation Error (G. T. E) associated with all the methods which is just the difference between the approximate solutions of the problem using the methods and the exact solutions of the problems.

GLOBAL TRUNCATION ERROR

| EULER'S | TAYLOR'S SERIES | HEUN'S | RUNGE – KUTTA |
|---|---|---|---|
| G. T. E at | G. T. E at | G. T. E at | G. T. E at |
| $x = 3$ | $x = 3$ | $x = 3$ | $x = 3$ |
| y(3) – ym | y(3) – ym | y(3) – ym | y(3) – ym |
| 0.065138 | - 0.0000023 | - 0.002879 | - 0.0000023 |

Euler's General steps,

$y_{k+1} = y_{k+h}\ f(x_k, y_k)$ for $k = 0,1, \ldots., m - 1$

which is an approximation of Taylor series expansion

$$y_{k+1} = y_{k+h}\ f(x_k, y_k) + \frac{h^2}{2!} f(x_k, y_k)$$

$$+ \ldots\ldots\ldots\ldots + \frac{hr}{r!} f^{(r)}(x_k, y_k)$$

has a local truncation Error (L. T. E.)

$$y^{(2)}(ck)\frac{h^2}{2!} \text{ for each of}$$

The neglected term which accumulated and made the (G. T. E) given by

$y(3) - ym = 0.065318$ to be very long. It has been proved that the error reduces by ½ when the step size is decreased by factor of ½.

When Taylor series is used to solve the problem under discussion $y^1 = \frac{x - y}{2}$ on (0, 3) at $h = 0.25$

The global truncation Error (G. T. E.) approximate to y(3) at $x = 3$ given by $y(3) - ym = - 0.0000023$ at $h = 0.25$.

Heun's method: The Global truncation error in solving the problem under discussion is of order $0(h^2)$.

It has been found also that the error decreases by about ¼ when the step size is reduced by factor of ½.

## 5.3 RECOMMENDATION

From the analysis above, it is clear that Runge – Kutta and Taylor's series methods performed better than Euler and Heun's method. It is also obvious that Heuns's method performed better than Euler's method.

Generally, the Euler can be reduced by reducing the step size.

In conclusion, therefore, for more accurate approximate solution of first order ordinary differential equation, we recommend Runge – Kutta and Taylor's series method, which is Taylor's method?

REFERENCE:

John H Mathews

Numerical Methods, Prentice Hall International Edition

William V Derrick, Stanley I. Grossman

Elements of Differential Equations with application Second Edition

Berg H .K Boebert W. E Franta W.A Moher

Formal Methods of Program and Verification,Prentice Hall, Englewood cliffs, New Jersey 07632

Margaret S Wu,

Introduction to Data Processing with basic, Harcourt Brace Jovanovick Inc. 1980

T

```
Private Sub cmdClearEntries_Click()
txtInterval(0) = ""
txtInterval(1) = ""
txth = ""
txtYInitial = ""

End Sub

Private Sub cmdClearTable_Click()
MSFlexGrid1.Clear
MSFlexGrid1.Rows = 1
End Sub

Private Sub cmdEnd_Click()
Unload Me
End Sub


Private Sub cmdPrint_Click()
On Error GoTo ending:

Form5.CommonDialog1.PrinterDefault = True
Form5.CommonDialog1.ShowPrinter
Call grPrint(MSFlexGrid1, "Numerical Solutions of Differential Equations",
"(Taylor's Method)", Val(txth))
Printer.EndDoc
MsgBox "Your Table Result has been sent to the printer"
Exit Sub

ending:

End Sub

Private Sub cmdSolve_Click()
On Error GoTo errhandler

Dim gr As MSFlexGrid
Dim y As Single
Dim x As Single

Set gr = MSFlexGrid1

gr.Rows = 1
gr.Col = 0
gr.ColWidth(2) = 2000



If IsNumeric(txtInterval(0)) = True And txtInterval(0) <> "" And
IsNumeric(txtInterval(1)) = True And txtInterval(1) <> "" Then

Dim interval As Integer
interval = (Val(txtInterval(1)) - Val(txtInterval(0))) / Val(txth)


For count1 = CInt(txtInterval(0)) To interval Step 1
```

```
    gr.TextMatrix(gr.Rows - 1, 0) = count1
    gr.TextMatrix(gr.Rows - 1, 1) = CInt(txtInterval(0)) + count1 * Val(txth)


    If gr.Rows > 1 Then
    Let y = Val(gr.TextMatrix(gr.Rows - 2, 2))
    Let x = Val(gr.TextMatrix(gr.Rows - 2, 1))

    Formula = (x - y) / 2
    d1 = (x - y) / 2
    d2 = (2 - x + y) / 4
    d3 = (-2 + x - y) / 8
    d4 = (2 - x + y) / 16
    H = Val(txth)
    gr.TextMatrix(gr.Rows - 1, 2) = Val(gr.TextMatrix(gr.Rows - 2, 2)) + (d1 *
H) + (d2 * H ^ 2) / 2 + (d3 * H ^ 3) / 6 + (d4 * H ^ 4) / 24
    Else
    gr.TextMatrix(gr.Rows - 1, 2) = Val(txtYInitial)

    End If

    gr.Rows = gr.Rows + 1
Next count1

End If
Exit Sub

errhandler:
MsgBox "Error In Entries"


End Sub
```

Ru

```
Private Sub cmdClearEntries_Click()
txtInterval(0) = ""
txtInterval(1) = ""
txth = ""
txtYInitial = ""

End Sub

Private Sub cmdClearTable_Click()
MSFlexGrid1.Clear
MSFlexGrid1.Rows = 1
End Sub

Private Sub cmdEnd_Click()
Unload Me
End Sub


Private Sub cmdPrint_Click()
On Error GoTo ending:

Form5.CommonDialog1.PrinterDefault = True
Form5.CommonDialog1.ShowPrinter
Call grPrint(MSFlexGrid1, "Numerical Solutions of Differential Equations",
"(Runge-Kutta Method)", Val(txth))
Printer.EndDoc
MsgBox "Your Table Result has been sent to the printer"
Exit Sub

ending:

End Sub

Private Sub cmdSolve_Click()
On Error GoTo errhandler

Dim gr As MSFlexGrid
Dim y As Single
Dim x As Single

Set gr = MSFlexGrid1

gr.Rows = 1
gr.Col = 0
gr.ColWidth(2) = 2000



If IsNumeric(txtInterval(0)) = True And txtInterval(0) <> "" And
IsNumeric(txtInterval(1)) = True And txtInterval(1) <> "" Then

Dim interval As Integer
interval = (Val(txtInterval(1)) - Val(txtInterval(0))) / Val(txth)


For count1 = CInt(txtInterval(0)) To interval Step 1
```

R

```
    gr.TextMatrix(gr.Rows - 1, 0) = count1
    gr.TextMatrix(gr.Rows - 1, 1) = CInt(txtInterval(0)) + count1 * Val(txth)


    If gr.Rows > 1 Then
    Formula = (x - y) / 2
    H = Val(txth)
    Let y = Val(gr.TextMatrix(gr.Rows - 2, 2))
    Let x = Val(gr.TextMatrix(gr.Rows - 2, 1))
   f1 = (x - y) / 2

    Let y = Val(gr.TextMatrix(gr.Rows - 2, 2)) + (H / 2) * f1
    Let x = Val(gr.TextMatrix(gr.Rows - 2, 1)) + (H / 2)
    f2 = (x - y) / 2

    Let y = Val(gr.TextMatrix(gr.Rows - 2, 2)) + (H / 2) * f2
    Let x = Val(gr.TextMatrix(gr.Rows - 2, 1)) + (H / 2)
    f3 = (x - y) / 2


    Let y = Val(gr.TextMatrix(gr.Rows - 2, 2)) + (H * f3)
    Let x = Val(gr.TextMatrix(gr.Rows - 2, 1)) + (H)
    f4 = (x - y) / 2


    gr.TextMatrix(gr.Rows - 1, 2) = Val(gr.TextMatrix(gr.Rows - 2, 2)) + (H / 6)
* (f1 + (2 * f2) + (2 * f3) + (f4))

    Else
    gr.TextMatrix(gr.Rows - 1, 2) = Val(txtYInitial)

    End If

    gr.Rows = gr.Rows + 1
Next count1

End If

Exit Sub

errhandler:
MsgBox "Error In Entries"


End Sub
```

```
Private Sub cmdClearEntries_Click()
txtInterval(0) = ""
txtInterval(1) = ""
txth = ""
txtYInitial = ""

End Sub

Private Sub cmdClearTable_Click()
MSFlexGrid1.Clear
MSFlexGrid1.Rows = 1
End Sub

Private Sub cmdEnd_Click()
Unload Me
End Sub


Private Sub cmdPrint_Click()
On Error GoTo ending:

Form5.CommonDialog1.PrinterDefault = True
Form5.CommonDialog1.ShowPrinter
Call grPrint(MSFlexGrid1, "Numerical Solutions of Differential Equations",
"(Heun's Method)", Val(txth))
Printer.EndDoc
MsgBox "Your Table Result has been sent to the printer"
Exit Sub

ending:

End Sub

Private Sub cmdSolve_Click()
On Error GoTo errhandler

Dim gr As MSFlexGrid
Dim y As Single
Dim x As Single

Set gr = MSFlexGrid1

gr.Rows = 1
gr.Col = 0
gr.ColWidth(2) = 2000




If IsNumeric(txtInterval(0)) = True And txtInterval(0) <> "" And
IsNumeric(txtInterval(1)) = True And txtInterval(1) <> "" Then

Dim interval As Integer
interval = (Val(txtInterval(1)) - Val(txtInterval(0))) / Val(txth)
```

```
For count1 = CInt(txtInterval(0)) To interval Step 1
    gr.TextMatrix(gr.Rows - 1, 0) = count1
    gr.TextMatrix(gr.Rows - 1, 1) = CInt(txtInterval(0)) + count1 * Val(txth)


    If gr.Rows > 1 Then

    H = Val(txth)
    Let y = Val(gr.TextMatrix(gr.Rows - 2, 2))
    Let x = Val(gr.TextMatrix(gr.Rows - 2, 1))
    Formula = (x - y) / 2
    Pk1 = y + (H * ((x - y) / 2))
    Tk1 = x + H

    Secondformula = (Tk1 - Pk1) / 2


    gr.TextMatrix(gr.Rows - 1, 2) = y + ((H / 2) * (Formula + Secondformula))
    Else
    gr.TextMatrix(gr.Rows - 1, 2) = Val(txtYInitial)

    End If

    gr.Rows = gr.Rows + 1
Next count1

End If
Exit Sub
errhandler:
MsgBox "Error In Entries"


End Sub
```