VISVESVARAYA TECHNOLOGICAL UNIVERSITY

JNANA SANGAMA, BELAGAVI-590018, KARNATAKA

A J INSTITUTE OF ENGINEERING & TECHNOLOGY

(A unit of Laxmi Memorial Education Trust. (R))
NH - 66, KottaraChowki, Kodical Cross - 575 006

Offering Department

DEPARTMENT OF MATHEMATICS

Beneficiary Department

DEPARTMENT OF MECHANICAL ENGINEERING

(Accredited by NBA)

MASTER MANUAL

Course: MATHEMATICS II FOR MECHANICAL ENGINEERING STREAM
Course Code: BMATM201

II SEMESTER

Prepared By:

Dr.ShanthaKumari K
Professor, Department of Mathematics

Academic Year: 2022-23

TABLE OF CONTENTS

VISION OF THE INSTITUTE

"To produce top-quality engineers who are groomed for attaining excellence in their profession and competitive enough to help in the growth of nation and global society."

MISSION OF THE INSTITUTE

M1: To offer affordable high-quality graduate program in engineering with value education and make the students socially responsible.

M2: To support and enhance the institutional environment to attain research excellence in both faculty and students and to inspire them to push the boundaries of knowledge base.

M3: To identify the common areas of interest amongst the individuals for the effective industry-institute partnership in a sustainable way by systematically working together.

M4: To promote the entrepreneurial attitude and inculcate innovative ideas among the engineering professionals.

VISION OF THE BENIFICIARY DEPARTMENT

"To create globally competent and self-reliant mechanical engineers adaptive to an interdisciplinary environment contributing to society through development, authority and entrepreneurship."

MISSION OF THE BENIFICIARY DEPARTMENT

M1: To offer high quality graduate program in the field of Mechanical Engineering with value education to the students and make them responsive to societal needs.

M2: To nurture the students with a global outlook for a sustainable future with high moral and ethical values.

M3: To strengthen collaboration with industries, academia and research organizations to enrich learning environment, thus enhance research and entrepreneurship culture.

M4: To create awareness about the need of interdisciplinary applications through alumni industry-institution interactions.

PROGRAM OUTCOMES (POs)

PO1: Engineering Knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

PO2: Problem Analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO3: Design/Development of Solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO4: Conduct Investigations of Complex Problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO5: Modern Tool Usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modelling to complex engineering activities with an understanding of the limitations.

PO6: The Engineer and Society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO7: Environment and Sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8: Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9: Individual and Team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO10:Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO11: Project Management and Finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12: Life-Long Learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

GENERAL LAB GUIDELINES

Do's

- Students should report to the concerned labs as per the given timetable.
- Students should make an entry in the log book whenever they enter the labs during practical.
- Students must bring Observation book, record and manual along with pen, pencil, and eraser etc., no borrowing from others.
- Students must use the computers carefully, as they are expensive. Each person may only use one computer at a time
- Any damage to the lab computers will be viewed seriously.
- Students may not install software on lab computers. If you have a question regarding specific software that you need to use, contact the concerned Faculty and Labs support team.
- When the experiment is completed, students should shut down the computers and make the counter entry in the logbook
- Wear your College ID card
- Avoid unnecessary talking while doing the experiment
- Do not panic if you do not get the output
- Students should not leave the lab without concerned faculty's permission.
- Before leaving the lab, students should check whether they have switched off the computers and the power supplies and kept their chairs properly.
- In each Lab student have to show the record of previous Lab.
- Each Lab will be evaluated for 15 marks and finally average will be taken for 15 marks.
- Viva questions shall be asked in labs and attendance also can be considered for everyday Lab evaluation.
- Tests shall be considered for 10 marks and final Lab assessment is for 25 marks.
- Student has to score minimum 10 marks out of 25 to pass Lab component.

Don'ts

- Do not come late to the Laboratory.
- Do not enter the laboratory without an ID card, lab dress code, observation book and record.
- Do not leave the laboratory without the permission of the faculty in-charge.
- Do not handle any equipment before reading the instructions/instruction manuals.
- Do not exchange the computers with others and hardware kits also.
- Do not misbehave in the laboratory.
- Do not alter computer settings/software settings.
- External Disk/drives should not be connected to computers without permission, doing so will attract fines.
- Do not mishandle the equipment / Computers.

- Usage of Mobile phones, tablets and other portable devices are not allowed in restricted places.

Rules for Maintaining the Record

- Write your name, USN and name of the subject on the outside front cover of the record. Write the same information in the first page inside the cover page.
- Update Table of Contents every time you start each new experiment or topic.
- Always use pen and write neatly and clearly
- Start each new topic (experiment number, experiment name, etc.) on a right-side (odd numbered) page.
- Obvious care should be taken to make it readable, even if you have bad handwriting
- Date to be written on every page on the top right-side corner
- Attach printouts and plots of data as needed.
- Strictly observe the instructions given by the Teacher/ Lab Instructor.

SYLLABUS

Mathematics-II for Mechanical Engineering stream (Effective from the academic year 2022 -2023) SEMESTER – II			
Course Code	**BMATM201**	CIE Marks	**50**
CREDITS	**04**	SEE Marks	**50**
Teaching Hours/Week(L:T:P: S)	**2:2:2:0**	Exam Hours	**3 hours**
Course Type : Integrated			
Total Hours of Pedagogy 40 hours Theory + 10 to12 Lab			

Course Objectives :

The goal of the course **Mathematics-II for Mechanical Engineering stream**(**BMATM201**) is to

- **Familiarize** the importance of Integral calculus and Vector calculus essential for Mechanical engineering.
- **Analyze** Mechanical engineering problems by applying Partial Differential Equations.
- **Develop** the knowledge of solving Mechanical engineering problems numerically.

Module-1: Integral Calculus (8 hours)

Introduction to Integral Calculus in Mechanical Engineering applications :

Multiple Integrals: Evaluation of double and triple integrals, evaluation of double integrals by change of order of integration, changing into polar coordinates. Applications to find Area and Volume by double integral. Problems.

Beta and Gamma functions: Definitions, properties, relation between Beta and Gamma functions. Problems.

Module-2: Series Expansion and Multivariable Calculus (8 hours)

Introduction to Vector Calculus in Mechanical Engineering applications.

Vector Differentiation: Scalar and vector fields. Gradient, directional derivative, curl and divergence - physical interpretation, solenoidal and irrotational vector fields. Problems.

Vector Integration: Line integrals, Surface integrals. Applications to work done by a force andflux. Statement of Green's theorem and Stoke's theorem. Problems.

Module-3: Ordinary Differential Equations (ODEs) of First Order (8 hours)

Importance of partial differential equations for Mechanical Engineering application.

Formation of PDE's by elimination of arbitrary constants and functions. Solution of nonhomogeneous PDE by direct integration. Homogeneous PDEs involving derivatives with respect to one independent variable only. Solution of Lagrange's linear PDE. Derivation of one-dimensional heat equation and wave equation..

Module-4: Numerical Methods (8 hours)

Importance of numerical methods for discrete data in the field of Mechanical Engineering.

Solution of algebraic and transcendental equations: Regula-Falsi and Newton-Raphson methods (only formulae). Problems.

Finite differences, Interpolation using Newton's forward and backward difference formulae, Newton's divided difference formula and Lagrange's interpolation formula (All formulae without proof). Problems.

Numerical integration: Trapezoidal, Simpson's (1/3)rd and (3/8)th rules(without proof). Problems.

Module-5: Numerical Methods-II (8 hours)

Introduction to various numerical techniques for handling Mechanical Engineering applications.

Numerical Solution of Ordinary Differential Equations (ODEs):

Numerical solution of ordinary differential equations of first order and first degree - Taylor's series method, Modified Euler's method, Runge-Kutta method of fourth order and Milne's predictor corrector formula (No derivations of formulae). Problems

- **Suggested Learning Resources:**

Books (Title of the Book/Name of the author/Name of the publisher/Edition and Year)

Text Books :

1. **B. S. Grewal** : "Higher Engineering Mathematics", Khanna Publishers, 44th Ed., 2021.
2. **E. Kreyszig**: "Advanced Engineering Mathematics", John Wiley & Sons, 10th Ed., 2018.

Reference Books :

1. **V. Ramana**: "Higher Engineering Mathematics" McGraw-Hill Education, 11th Ed., 2017
2. **Srimanta Pal &SubodhC.Bhunia**: "Engineering Mathematics" Oxford University

Press, 3rd Ed., 2016.

3. **N.P Bali and Manish Goyal**: "A Textbook of Engineering Mathematics" Laxmi Publications, 10thEd., 2022.
4. **C. Ray Wylie, Louis C. Barrett**: "Advanced Engineering Mathematics" McGraw – Hill Book Co., New York, 6th Ed., 2017.
5. **Gupta C.B, Sing S.R and Mukesh Kumar:** "Engineering Mathematic for Semester I and II", Mc-Graw Hill Education(India) Pvt. Ltd 2015.
6. **H.K. Dass and Er. RajnishVerma**: "Higher Engineering Mathematics" S. Chand Publication, 3rd Ed., 2014.
7. **James Stewart**: "Calculus" Cengage Publications, 7thEd., 2019

Web links and Video Lectures (e-Resources):

- http://nptel.ac.in/courses.php?disciplineID=111
- http://www.class-central.com/subject/math(MOOCs)
- http://academicearth.org/
- VTU e-Shikshana Program
- VTU EDUSAT Program

Course outcomes :

At the end of the course the student will be able to:

CO1	Apply the knowledge of multiple integrals to compute area and volume.
CO2	Understand the applications of vector calculus refer to solenoidal, irrotational vectors, line integral and surface integral.
CO3	Demonstrate partial differential equations and their solutions for physical interpretations
CO4	Apply the knowledge of numerical methods in solving physical and engineering phenomena.
CO5	Solve first order ordinary differential equations arising in engineering problems
CO6	Familiarize with modern mathematical tool - PYTHON

Mapping of Course Outcomes with POs

Course Outcomes (COs)	**Program Outcomes (POs)**											
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
CO1	3	2	1		2							
CO2	3	2	1		2							
CO3	3	2	1		2							
CO4	3	3	1		3							
CO5	3	3	1		3							
CO6	3	3	1		3							
Average												

ASSESSMENT DETAILS (BOTH CIE AND SEE)

- The weightage of Continuous Internal Evaluation (CIE) is 50% and for Semester End Exam (SEE) is 50%.
- The minimum passing mark for the CIE is 40% of the maximum marks (20 marks).
- A student shall be deemed to have satisfied the academic requirements and earned the credits allotted to each subject/course if the student secures not less than 35% (18 Marks out of 50) in the semester-end examination(SEE), and a minimum of 40% (40 marks out of 100) in the sum total of the CIE (Continuous Internal Evaluation) and SEE (Semester End Examination) taken together.

CONTINUOUS INTERNAL EVALUATION (CIE):

❖ **Two Unit Tests** each of **50Marks scaled down to 25 Marks(Duration 1.5 hours**)

1. First test at the end of 6^{th} week of the semester
2. Second test at the end of the15^{th} week of the semester
3. Average of two internal tests is taken for 25 Marks and scaled down to 15 marks. The minimum marks to be secured in Internal Tests to appear for SEE shall be 6 (40% of maximum marks-15)

❖ **One Assignment and One Quiz** each of **10 Marks**

1. First assignment/Quiz at the end of 4^{th} week of the semester
2. Second assignment/Quiz at the end of 9^{th} week of the semester
3. Total 20 marks obtained in these two assessment methods will be scaled down to 10 Marks. The minimum marks to be secured in Assignments + Quiz to appear for SEE shall be 4 (40% of maximum marks-10)

❖ Total Theory CIE marks are Scaled down Marks of Tests and Assignments to 25. The minimum marks to be secured in Tests and Assignments to appear for SEE shall be 10 (40% of maximum marks-25)

❖ **CIE for the practical component of the IC**

1. On completion of every experiment/program in the laboratory, the students shall be evaluated and marks shall be awarded on the same day. The 15 marks are for

conducting the experiment and preparation of the laboratory record, the other 10 marks shall be for the test conducted at the end of the semester. Marks of all experiments, write-ups are added and scaled down to 15 marks. The minimum marks to be secured in conducting the experiment and preparation of the laboratory record to appear for SEE shall be 06 (40% of maximum marks-15)

2. The laboratory test (**duration 03 hours**) at the end of the 15th week of the semester/after completion of all the experiments (whichever is early) shall be conducted for 50 marks and scaled down to 10 **marks**. The minimum marks to be secured in laboratory test to appear for SEE shall be 04 (40% of maximum marks-10)
3. Scaled-down marks of experiments, record and tests added will be CIE marks for the laboratory component of IC/IPCC for **25 marks**. The minimum marks to be secured in laboratory test to appear for SEE shall be 10 (40% of maximum marks-25)
4. The minimum marks to be secured in CIE to appear for SEE shall be 10 (40% of maximum marks-25) in the theory component and 10 (40% of maximum marks-25) in the practical component. The laboratory component of the IC/IPCC shall be for CIE only. However, in SEE, the questions from the laboratory component shall be included. The maximum of 05 questions is to be set from the practical component of IC/IPCC, the total marks of all questions should not be more than 25 marks. The theory component of the IC shall be for both CIE and SEE.

Split up of Marks	**Practical Sessions- Continuation Evaluation (CE) Methodology / Process Steps per Experiment**	**Marks**
#R1	Observation, Write up of Procedure / Algorithm/ Program and Execution of experiment	5
#R2	Record writing	5
#R3	Viva – Voce (Questions & Answers on relevant Experiment /Topic)	5
	Total Marks for each experiment	**15**
	Practical Sessions-Internal Assessment (IA) **(**conducted for 50 marks and scaled down to 10 **marks)**	
#R1	Write-up of Procedure/Program/Algorithm	20
#R2	Conduction/Execution	20
#R3	Viva-Voce	10
	Total Marks	**50**

❖ **Semester End Examination (SEE) :**

Theory SEE will be conducted by University as per the scheduled timetable, with common question papers for the subject (**duration 03 hours**)

- The question paper shall be set for 100 marks. The medium of the question paper shall be English/Kannada). The duration of SEE is 03 hours.
- The question paper will have 10 questions. Two questions per module. Each question is set for 20 marks. The students have to answer 5 full questions, selecting one full question from each module. The student has to answer for 100 marks and **marks scored out of 100 shall be proportionally reduced to 50 marks.** The minimum marks to be secured in SEE is 18 (35% of maximum marks-50)
- There will be 2 questions from each module. Each of the two questions under a module (with a maximum of 3 sub-questions), should have a mix of topics under that module.
- The students have to answer 5 full questions, selecting one full question from each module

List of Laboratory experiments (2 hours/week per batch/ batch strength 15) 10 lab sessions + 1 repetition class + 1 Lab Assessment

Lab 1: Program to compute surface area, volume and centre of gravity.

Lab 2: Evaluation of improper integrals, Beta und Gamma functions.

Lab 3: Finding gradient, divergent, curl and their geometrical interpretation.

Lab 4: Verification of Green's theorem

Lab 5: Solution of Lagrange's linear partial differential equations

Lab 6 : Solution of algebraic and transcendental equations by Regula-Falsi and Newton-Raphson method

Lab 7: Interpolation/Extrapolation using Newton's forward and backward difference formula

Lab 8: Computation of area under the curve using Trapezoidal, Simpson's (1/3)rd and (3/8)th rule

Lab 9:Solution of ODE of first order and first degree by Taylor's series and Modified Euler's method

Lab 10: Solution of ODE of first order and first degree by Runge-Kutta 4th order and Milne's predictor-corrector method

Suggested software's : Mathematica/MatLab/Python/Scilab

LAB 1: Programme to compute area, volume and center of gravity

1.1 Objectives:

Use python

1. to evaluate double integration.

2. to compute area and volume.

Syntax for the commands used:

1. pprint ()

2. integrate:

integrate (function ,(variable , min_limit , max_limit))

1.2 Double and triple integration :

Example 1: Evaluate the integral $\int_0^1 \int_0^x (x^2 + y^2) dydx$

Code :

```
fromsympyimport *
x,y,z= symbols ('x y z')
w1= integrate (x ** 2+y ** 2 ,(y,0,x) ,(x,0,1))
print(w1)
```

Output :

1/3

Example 2: Evaluate the integral $\int_0^3 \int_0^{3-x} \int_0^{3-x-y} (xyz) dzdydx$

Code :

```
from sympy import *
x= Symbol ('x')
y= Symbol ('y')
z= Symbol ('z')
w2= integrate ((x*y*z) ,(z,0,3-x-y) ,(y,0,3-x) ,(x,0,3))
print(w2)
```

Output :

81/80

Example 3: Prove that $\iint (x^2 + y^2)dydx = \iint (x^2 + y^2)dxdy$

Code :

```
from sympy import *
x= Symbol ('x')
y= Symbol ('y')
z= Symbol ('z')
w3= integrate (x ** 2+y ** 2,y,x)
w4= integrate (x ** 2+y ** 2,x,y)
print("LHS =")
display(w3)
print("RHS =")
display(w4)
```

Output :

LHS =

$$\frac{x^3y}{3} + \frac{xy^3}{3}$$

RHS =

$$\frac{x^3y}{3} + \frac{xy^3}{3}$$

1.3 Area and Volume

Area of the region R in the cartesian form is $\int_R \int dxdy$

Example 4:

Find the area of an ellipse by double integration. $A = 4\int_0^{a/b/a)} \sqrt{a^2 - x^2}dydx$

Code :

```
from sympy  import *
x= Symbol ('x')
y= Symbol ('y')
a=4
b=6
w3=4* integrate (1 ,(y,0 ,(b/a)* sqrt(a ** 2-x ** 2)) ,(x,0,a))
print(w3)
```

Output :

24.0*pi

1.4 Area of the region **R** in the polar form is $\int_R \int r dr d\theta$

Example 5:
Find the area of the cardioid $r = a(1 + \cos\theta)$ by double integration

Code :

```
from sympy import *
r= Symbol ('r')
t= Symbol ('t')
a= Symbol ('a')
w3=2* integrate (r ,(r,0,a*(1+cos (t))) ,(t,0,pi))
pprint (w3)
```

Output : $\frac{3\pi a^2}{2}$

1.5 Volume of a solid is given by $\iiint dxdydz$

Example 6:
Find the volume of the tetrahedron bounded by the planes $x = 0, y = 0$ and $z = 0,$ and

$$\frac{x}{a} + \frac{y}{b} + \frac{z}{c} = 1$$

Code :

```
from sympy import *
x= Symbol ('x')
y= Symbol ('y')
z= Symbol ('z')
a= Symbol ('a')
b= Symbol ('b')
c= Symbol ('c')
w2= integrate (1 ,(z,0,c*(1-x/a-y/b)) ,(y,0,b*(1-x/a)) ,(x,0,a))
print(w2)
```

Output :

a*b*c/6

1.6 Exercise:

1. Evaluate $\int_0^1 \int_0^x (x + y) dydx$
   Ans: 0.5
2. Find the $\int_0^{\log(2)} \int_0^x \int_0^{x+\log(y)} (e^{x+y+z}) dzdydx$
   Ans: -0.2627
3. Find the area of positive quadrant of the circle $x^2 + y^2 = 16$
   Ans: 4π

4 Find the volume of the tetrahedron bounded by the planes $x = 0, y = 0$ and $z = 0$, $\frac{x}{2} + \frac{y}{3} + \frac{z}{4} = 1$
Ans: 4

Viva Questions :

1. What is the syntax for integration in Python ?
   Ans : integrate (function ,(variable , min_limit , max_limit))

2. What is the formula for Area using double integration?
   Ans : $\int \int dx\, dy$

3. What is the formula for Area using double integration in polar form?
   Ans : $\int \int r\ dr\, d\theta$

4. What is the formula for Volume using double integrals?
   Ans : $\int \int z\ dx\, dy$

5. What is the use of pprint command?
   Ans : For Pretty Print the Mathematical expressions

LAB 2: Evaluation of improper integrals, Beta and Gamma functions

2.1 Objectives:

Use python

1 to evaluate improper integrals using Beta function.

2 to evaluate improper integrals using Gamma function.
Syntax for the commands used:

1. gamma

math .gamma (x)

Parameters :
x : The number whose gamma value needs to be computed.

2. beta

math .beta (x,y)

Parameters :
x ,y: The numbers whose beta value needs to be computed.
Note: We can evaluate improper integral involving infinity by using inf.

Example 1: Evaluate $\int_0^\infty e^{-x}dx$

Code :

```
from sympy import *
x= symbols ('x')
w1= integrate (exp(-x) ,(x,0, float ('inf ')))
print( simplify (w1))
```

Output :1

Gamma function is $\Gamma(n) = \int_0^\infty e^{-x}x^{n-1}dx$

Example 2:

Evaluate $\Gamma(5)$ by using definition

Code :

```
from sympy import *
x= symbols ('x')
w1= integrate (exp(-x)*x ** 4 ,(x,0, float ('inf')))
print( simplify (w1))
```

Output :24

Example 3 : Find Beta(3,5), Gamma(5)

Code :

```
# beta and gamma functions
from sympy import beta , gamma
m= input ('m :');
n= input ('n :');
m= float (m);
n= float (n);
s= beta (m,n);
t= gamma (n)
print('gamma (',n,') is %3.3f '%t)
print('Beta (',m,n,') is %3.3f '%s)
```

Output :

```
m :3
n :5
gamma ( 5.0 ) is 24.000
Beta ( 3.0 5.0 ) is 0.010
```

Example 4: Calculate Beta(5/2,7/2) and Gamma(5/2).

Code :

```
# beta and gamma functions
# If the number is a fraction give it in decimals .Eg 5/2=2.5
from sympy import beta , gamma
m= float ( input('m : '));
n= float ( input('n :'));
s= beta (m,n);
t= gamma (n)
print('gamma (',n,') is %3.3f '%t)
print('Beta (',m,n,') is %3.3f '%s)
```

Output :

```
m : 2.5
n :3.5
gamma ( 3.5 ) is 3.323
Beta ( 2.5 3.5 ) is 0.037
```

Example 5 :
Verify that Beta(m, n) = Gamma(m)Gamma(n)/Gamma(m + n) for m=5 and n=7

Code :

```
from sympy import beta , gamma
m=5;
n=7;
m= float (m);
n= float (n);
s= beta (m,n);
```

```
t=( gamma (m)* gamma (n))/ gamma (m+n);
print(s,t)
if(abs (s-t)<=0. 00001 ):
        print('beta and gamma are related ')
else :
        print('given values are wrong ')
```

Output :

0.000432900432900433 0.000432900432900433
beta and gamma are related

2.2 Exercise:

1 Evaluate $\int_0^{\infty} e^{-t}\cos(2t)dt$
Ans: $1/5$

2 Find the value of $\text{Beta}(5/2,9/2)$
Ans: 0.0214

3 Find the value of Gamma(13)
Ans: 479001600

4 Verify that $\text{Beta}(m,n) = \text{Gamma}(m)\text{Gamma}(n)/\text{Gamma}(m+n)$ for $m = 7/2$ and $\mathbf{n} = 11/2$
Ans: True

Viva Questions :

1. What is the syntax to evaluate Gamma function in Python?

   Ans :math .gamma (x)

2. What is the syntax to evaluate Beta function in Python?

   Ans :math .beta (x,y)

3. What is the relation between Beta and Gamma function?
   Ans : $\boldsymbol{B(m,n) = \frac{\Gamma(m)\Gamma(n)}{\Gamma(m+n)}}$
4. What is the value of $\Gamma(5)$?
   Ans : $\boldsymbol{4! = 24}$
5. What is the value of Gamma$(\frac{1}{2})$?
   Ans : $\boldsymbol{\sqrt{\pi}}$

LAB 3: Finding gradient, divergent, curl and their geometrical interpretation

3.1 Objectives:
Use python

1. to find the gradient of a given scalar function.
2. to find find divergence and curl of a vector function.

Example 1 : Find gradient of a scalar point function x^2yz

Code :

```
find gradient of  F = x^2yz
#To find gradient of a scalar point function x^2yz
from sympy.physics.vector import *
from sympy import var
var('x,y,z')
v=ReferenceFrame('v')
F=v[0]**2*v[1]*v[2]
G=gradient(F,v)
F=F.subs([(v[0],x),(v[1],y),(v[2],z)])
print("Given scalar function F=")
display(F)
G=G.subs([(v[0],x),(v[1],y),(v[2],z)])
print("\n Gradient of F=")
display(G)
```

Output :

Given scalar function $F =$

$$x^2yz$$

Gradient of F $=$

$$2xyz\hat{\mathbf{v}}_{\mathbf{x}} + x^2z\hat{\mathbf{v}}_{\mathbf{y}} + x^2y\hat{\mathbf{v}}_{\mathbf{z}}$$

Example 2 : To find divergence of $\vec{F} = x^2y\hat{\imath} + yz^2\hat{\jmath} + x^2z\hat{k}$.

Code :

```
from sympy.physics.vector import *
from sympy import var
var('x,y,z')
v=ReferenceFrame('v')
F=v[0]**2*v[1]*v.x+v[1]*v[2]**2*v.y+v[0]**2*v[2]*v.z
G=divergence(F,v)
```

```
F=F.subs([(v[0],x),(v[1],y),(v[2],z)])
print("Given vector point function is ")
display(F)
G=G.subs([(v[0],x),(v[1],y),(v[2],z)])
print("Divergence of F=")
display(G)
```

Output:

Given vector point function is

$$x^2y\hat{\mathbf{v}}_{\mathbf{x}} + yz^2\hat{\mathbf{v}}_{\mathbf{y}} + x^2z\hat{\mathbf{v}}_{\mathbf{z}}$$

Divergence of F =

$$x^2 + 2xy + z^2$$

Example 3 : To find curl of $\vec{F} = xy^2\hat{\imath} + 2x^2yz\hat{\jmath} - 3yz^2\hat{k}$

Code :

```
#To find curl of F = xy^2 i + 2x^2yz j - 3yz^2 k
from sympy.physics.vector import *
from sympy import var
var('x,y,z')
v=ReferenceFrame('v')
F=v[0]*v[1]**2*v.x+2*v[0]**2*v[1]*v[2]*v.y-3*v[1]*v[2]**2*v.z
G=curl(F,v)
F=F.subs([(v[0],x),(v[1],y),(v[2],z)])
print("Given vector point function is ")
display(F)
G=G.subs([(v[0],x),(v[1],y),(v[2],z)])
print("curl of F=")
display(G)
```

Output :

Given vector point function is

$$xy^2\hat{\mathbf{v}}_{\mathbf{x}} + 2x^2yz\hat{\mathbf{v}}_{\mathbf{y}} - 3yz^2\hat{\mathbf{v}}_{\mathbf{z}}$$

curl of F =

$$(-2x^2y - 3z^2)\hat{\mathbf{v}}_{\mathbf{x}} + (4xyz - 2xy)\hat{\mathbf{v}}_z$$

3.3 Exercise:

1 If $u = x + y + z, v = x^2 + y^2 + z^2, w = yz + zx + xy$, find grad u, grad v and grad w. Ans: $\hat{\imath} + \hat{\jmath} + \hat{k}, 2(x\hat{\imath} + y\hat{\jmath} + z\hat{k}), (y + z)\hat{\imath} + (z + x)\hat{\jmath} + (z + x)\hat{k}$.

2 Evaluate div F and curl F at the point (1,2,3), given that $\vec{F} = x^2yz\hat{\imath} + xy^2z\hat{\jmath} + xyz^2\hat{k}$. Ans: $6xyz, x(z^2 - y^2)\hat{\imath} + y(x^2 - z^2)\hat{\jmath} + z(y^2 - x^2)\hat{k}$

3 Prove that the vector $(yz - x^2)\hat{\imath} + (4y - z^2x)\hat{\jmath} + (2xz - 4z)\hat{k}$ is solenoidal.

4 Find the vector normal to the surface $xy^3z^2 = 4$ at the point $(-1, -1,2)$. Ans: $-4\hat{\imath} - 12\hat{\jmath} + 4\hat{k}$.

5 If $\vec{R} = x\hat{\imath} + y\hat{\jmath} + z\hat{k}$, show that (i) $\nabla \cdot \vec{R} = 3$, (ii) $\nabla \times \vec{R} = 0$.

Viva Questios :

1. What is the formula for Gradient of a scalar function?
   Ans : $\nabla \phi = \frac{\partial \phi}{\partial x}\, i + \frac{\partial \phi}{\partial y}\, j + \frac{\partial \phi}{\partial z}\, k$
2. What is the formula for Divergence of a vector function?
   Ans : $\nabla . \vec{A}$
3. What is the formula for Curl of a Vector function?
   Ans : $\nabla \times \vec{A}$
4. The gradient of a scalar function is a scalar or vector?
   Ans : Gradient is a vector function
5. The divergence of a vector function is a scalar or vector?
   Ans : divergence is a scalar function
6. Curl of a vector function is a scalar or vector?
   Ans : curl is is a vector function
7. ***divergence*** of a ***vector*** field is not a ***vector*** field, ***but*** a ***scalar*** . True or False?
   Ans : True
8. A vector field with a vanishing curl is called ---------
   Ans : Irrotational
9. A vector field with a vanishing divergence is called ---------

   Ans : Solenoidal

LAB 4: Verification of Green's theorem

1.1 Objectives:

- Use python to evaluate integrals using Green's theorem.

  Statement of Green's theorem in the plane: If $P(x,y)$ and $Q(x,y)$ be two continuous functions having continuous partial derivatives in a region R of the xy-plane, bounded by a simple closed curve C, then

$$\oint (Pdx + Qdy) = \iint_R \left(\frac{\partial Q}{\partial x} - \frac{\partial P}{\partial y}\right) dxdy.$$

Example 1: Using Green's theorem, evaluate, $\oint_c[(x+2y)dx + (x-2y)dy]$ where c is the region bounded by coordinate axes and the line $x = 1 and\ y = 1$.

Code :

```
from sympy import *
var('x,y')
M=x+2*y
N=x-2*y
f=diff(N,x)-diff(M,y)
soln=integrate(f,[x,0,1],[y,0,1])
print("I=",soln)
```

Output: I= -1

Example 2 : Using Green's theorem, evaluate, $\oint_c\ [(xy+y^2)dx + (x^2)dy]$ where c is the closed curve bounded by $y = x\ and\ y = x^2$.

Code :

```
fromsympy import *
var('x,y')
p=x*y+y**2
q=x**2
f=diff(q,x)-diff(p,y)
soln=integrate(f,[y,x**2,x],[x,0,1])
print("I=",soln)
```

Output: I= -1/20

Example 3 : Using Green's theorem, evaluate, $\oint_c\ [(3x+4y)dx + (2x-3y)dy]$, where c is the boundary of the circle $x^2 + y^2 = 4$.

Code :

```
from sympy import *
var('x,y')
p=3*x+4*y
q=2*x-3*y
f=diff(q,x)-diff(p,y)
soln=integrate(f,[y,-sqrt(4-x**2),sqrt(4-x**2)],[x,-2,2])
print("I=",soln)
```

Output: I= -8*pi

1.3 Exercise:

1 Using Green's theorem, evaluate $\oint [(3x + 4y)dx + (2x - 3y)dy]$, where c is the boundary of the circle $x^2 + y^2 = 4$.
Ans: -8π

Viva Questions :

1. What is a simple closed curve?

Ans :A simple closed curve is a curve which does not cross itself and has no endpoints.

2. State Greens theorem

Ans : If $P(x, y)$ and $Q(x, y)$ be two continuous functions having continuous partial derivatives in a region R of the xy-plane, bounded by a simple closed curve C, then

$$\oint (Pdx + Qdy) = \iint_R \left(\frac{\partial Q}{\partial x} - \frac{\partial P}{\partial y}\right) dxdy.$$

3. The path traversal in calculating the Green's theorem is –

Ans : Anticlockwise

4. The Green's theorem gives the relationship between --------

Ans : Relationship between a line integral and a surface integral

LAB 5: Solution of Lagrange's linear partial differential equations

5.1 Objectives:

Use python to solve linear Partial Differential Equations of first order

Example 1: Solve the PDE, $xp + yq = z$, where $z = f(x, y)$

Code :

```
from sympy import *
f = Function ('f')
z = f(x, y)
zx= z. diff (x)
zy= z. diff (y)
eq= Eq(x*zx+y*zy , z)                    # Solve xp+yq=z
pprint (eq)
print("\n")
soln= pdsolve (eq ,z)
display ( soln )
```

Output :

$$x\cdot\frac{\partial}{\partial x}(f(x, y)) + y\cdot\frac{\partial}{\partial y}(f(x, y)) = f(x, y)$$

$$f(x, y) = xF\left(\frac{y}{x}\right)$$

Example 2 : Solve the PDE $2p + 3q = 1$, where $p = \frac{\partial z}{\partial x}$ and $p = \frac{\partial z}{\partial y}$

Code :

```
from sympy import *
f = Function ('f')
z = f(x, y)
zx= z. diff (x)
zy= z. diff (y)
eq= Eq(2*zx+3*zy , 1)        # Solve 2p+3q=1
pprint (eq)
print("\n")
soln= pdsolve (eq ,z)
pprint ( soln )
```

Output :

$$2 \cdot \frac{\partial}{\partial x}(f(x,y)) + 3 \cdot \frac{\partial}{\partial y}(f(x,y)) = 1$$

$$f(x,y) = \frac{2 \cdot x}{13} + \frac{3 \cdot y}{13} + F(3 \cdot x - 2 \cdot y)$$

Example 3: Solve the PDE $x^2p + y^2q = (x+y)z$, where $p = \frac{\partial z}{\partial x}$ and $p = \frac{\partial z}{\partial y}$

Code :

```
from sympy . solvers .pdeimportpdsolve
from sympy import Function , Eq ,cot , classify_pde , pprint
from sympy .abcimport x, y, a
f = Function ('f')
z = f(x, y)
zx= z. diff (x)
zy= z. diff (y)
# Solve x^2p+y^2q =(x+y)z
eq=Eq(x ** 2*zx+y** 2*zy ,(x+y)*z)
pprint (eq)
print("\n")
soln= pdsolve (eq ,z)
pprint ( soln )
```

Output :

$$x\frac{\partial}{\partial x}(f(x,y)) + y^2 \cdot \frac{\partial}{\partial y}(f(x,y)) = (x+y) \cdot f(x,y)$$

$$f(x,y) = (x-y) \cdot f\left(\frac{-x+y}{x \cdot y}\right)$$

5.2 Exercise:

1. Solve $y^2p + x^2q = y^2x$
   Ans: $z = x^2/2 + F(y^3 - x^3)$
2. Solve $xp + yq = 3z$
   Ans: $z = x^3F(y/x)$
3. Solve $y^2p - xyq = x(z - 2y)$
   Ans: $x^2 + y^2 = F(y^2 - yz)$

Viva Questions :

1. What is the general form of Lagranges Linear partial differential equation?
   Ans : $Pp + Qq = R$, where P, Q and R are functions of x,y,z and $p = \frac{\partial z}{\partial x}$, $q = \frac{\partial z}{\partial y}$
2. What is the use of solvers module in Sympy ?

3. **Ans :**The *solvers* module in SymPy implements methods for solving equations.
4. What is the use of **classify_pde()** in python?
   Ans : It Classifies PDEs into possible hints for dsolve().
5. What is the use of **pdsolve()**
   Ans : It solves PDE's

LAB 6: Solution of algebraic and transcendental equation by Regula-Falsi and Newton-Raphson method

6.1 Objectives:

Use python

1 to solve algebraic and transcendental equation by Regula-Falsi method.

2 to solve algebraic and transcendental equation by Newton-Raphson method.

6.2 Regula-Falsi method to solve a transcendental equation

Example 1 : Obtain a root of the equation $x^3 - 2x - 5 = 0$ between 2 and 3 by Regula-falsi method. Perform 5 iterations.

Code :

```
from sympy import *
x= Symbol ('x')
g = input ('Enter the function ')  # %x^3-2*x-5;  % function
f= lambdify (x , g )
a= float ( input ('Enter a valus :') ) #2
b= float ( input ('Enter b valus :') ) # 3
N=int( input ('Enter number of iterations :') ) #5
for i in range (1 , N+1 ):
   c=( a*f ( b )-b*f ( a ) )/( f ( b )-f ( a ) )
   if (( f ( a )*f ( c )<0 ) ):
      b=c
   else :
      a=c
   print ('itration %d \t the root %0.3f \t function value %0.3f \n'%(i ,c , f ( c ) ) ) ;
```

Output :

```
Enter the function  x**3-2*x-5
Enter a valus :2
Enter b valus :3
Enter number of iterations :5
itration 1        the root 2.059         function value -0.391

itration 2        the root 2.081         function value -0.147

itration 3        the root 2.090         function value -0.055

itration 4        the root 2.093         function value -0.020

itration 5        the root 2.094         function value -0.007
```

6.3 Newton-Raphson method to solve a transcendental equation

Example 2 : Find a root of the equation 3x = cosx+1, near 1, by Newton Raphson method. Performiterations

Code :

```
from sympy import *
x= Symbol ('x')
g = input ('Enter the function ') #%3x -cos(x)-1; % function
f= lambdify (x , g )
dg = diff ( g ) ;
df= lambdify (x , dg )
x0= float ( input ('Enter the intial approximation ') ) ; # x0=1
n= int( input ('Enter the number of iterations ') ) ; #n=5;
for i in range (1 , n+1 ):
   x1 =( x0 - ( f ( x0 )/df ( x0 ) ) )
   print ('itration %d \t the root %0.3f \t function value %0.3f \n'% (i , x1 , f ( x1 ) ) ) ;
   # Above line is to print all iteration values
   x0 = x1
```

Output :

```
Enter the function  3*x-cos(x)-1
Enter the intial approximation  1
Enter the number of iterations  5
itration 1        the root 0.620         function value 0.046

itration 2        the root 0.607         function value 0.000

itration 3        the root 0.607         function value 0.000

itration 4        the root 0.607         function value 0.000

itration 5        the root 0.607         function value 0.000
```

6.4 Exercise:

1. Find a root of the equation $3\,x = cos\,x + 1$, between 0 and 1 , by Regula-falsi method. Perform 5 iterations.

Ans: 0.607

2. Find a root of the equation $x\,e^x = 2$, between 0 and 1 , by Regula-falsi method. Correct to 3 decimal places.

Ans: 0.853

3. Obtain a real positive root of $x^4 - x = 0$, near 1 , by Newton-Raphson method. Perform 4 iterations.

Ans: 1.856

4. Obtain a real positive root of $x^4 + x^3 - 7\,x^2 - x + 5 = 0$, near 3 , by Newton-Raphson method. Perform 7 iterations.

Ans: 2.061

Viva Questions :

1. The Newton Raphson method fails if _____
   Ans : $f'(x_0) = 0$
2. The Newton Raphson formula is ----------
3. **Ans :** $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$, $n = 0,1,2$ ...
4. RegulaFalsi formula is ---------
   Ans : $\boldsymbol{x = \frac{a\,f(b) - b\,f(a)}{f(b) - f(a)}}$
5. If $f(a)$ and $f(b)$ are of opposite signs then there exists a root of $f(x) = 0$ in the interval (a, b). True or False?
   Ans : True
6. The equation $xe^x - \cos x = 0$ is an Algebraic equation. True or False?
   Ans : False. It is a transcendental equation.

LAB 7: Interpolation /Extrapolation using Newton's forward and backward difference formula

7.1 Objectives:

Use python
1. to interpolate using Newton's Forward interpolation method.
2. to interpolate using Newton's backward interpolation method.
3. to extrapolate using Newton's backward interpolation method.

Example : Use Newton's forward interpolation to obtain the interpolating polynomial and hence calculate y(2) for the following:

x	1	3	5	7	9
y	6	10	62	210	502

Code :

```
from sympy import *
import numpy as np
n = int( input ('Enter number of data points : ') )
x = np . zeros (( n ) )
y = np . zeros (( n , n ) )
# Reading data points
print ('Enter data for x and y: ')
for i in range ( n ):
    x[i] = float ( input ( 'x['+str( i )+']= ') )
    y[i][0] = float ( input ( 'y['+str( i )+']= ') )
# Generating forward difference table
for i in range (1 , n ):
    for j in range (0 , n-i ):
        y[j][i] = y[j+1][i-1] - y[j][i-1]
print ('\n FORWARD DIFFERENCE TABLE \n') ;
for i in range (0 , n ):
    print ('%0.2f ' %( x[i]) , end='')
    for j in range (0 , n-i ):
        print ('\t\t%0.2f ' %( y[i][j]) , end='')
    print ()
# obtaining the polynomial
t= symbols ('t')
f=[]                                    # f is a list type data
p=( t-x[0])/( x[1]-x[0])
```

```
f . append ( p )
for i in range (1 , n-1 ):
    f . append ( f[i-1]*( p-i )/( i+1 ) )
    poly =y[0][0]
for i in range ( n-1 ):
    poly = poly +y[0][i+1]*f[i]
simp_poly = simplify ( poly )
print ('\n THE INTERPOLATING POLYNOMIAL IS\n') ;
pprint ( simp_poly )
# if you want to interpolate at some point the next session will help
inter = input ('Do you want to interpolate at a point (y/n)? ') # y
if inter =='y':
    a= float ( input ('enter the point ') ) #2
    interpol = lambdify (t , simp_poly )
    result = interpol ( a )
    print ('\n The value of the function at ', a ,'is \n', result ) ;
```

Output :

```
Enter number of data points: 5
Enter data for x and y:
x[0]=1
y[0]=6
x[1]=3
y[1]=10
x[2]-5
y[2]=62
x[3]-7
y[3]-210
x[4]-9
y[4]-502

FORWARD DIFFERENCE TABLE

1.00            6.00            4.00            48.00           48.00           0.00
3.00            10.00           52.00           96.00           48.00
5.00            62.00           148.00          144.00
7.00            210.00          292.00
9.00            502.00

THE INTERPOLATING POLYNOMIAL IS

     3       2
1.0⋅t  - 3.0⋅t  + 1.0⋅t + 7.0
Do you want to interpolate at a point(y/n)? y
enter the point 2

The  value of the function at 2.0 is
 5.0
```

Example 2 : Use Newtons backward interpolation to obtain the interpolating polynomial and hence calculate y(8) for the following data:

x	1	3	5	7	9
y	6	10	62	210	502

Code :

```
from sympy import *
import numpy as np
import sys
print (" This will use Newton 's backword intepolation formula ")
# Reading number of unknowns
n = int( input ('Enter number of data points : ') )
# Making numpy array of n & n x n size and initializing
# to zero for storing x and y value along with differences of y
x = np . zeros (( n ) )
y = np . zeros (( n , n ) )
# Reading data points
print ('Enter data for x and y: ')
for i in range ( n ):
    x[i] = float ( input ( 'x['+str( i )+']= ') )
    y[i][0] = float ( input ( 'y['+str( i )+']= ') )
# Generating backward difference table
for i in range (1 , n ):
    for j in range ( n-1 , i-2 ,-1 ):
        y[j][i] = y[j][i-1] - y[j-1][i-1]
print ('\ nBACKWARD DIFFERENCE TABLE \n') ;
for i in range (0 , n ):
    print ('%0.2f ' %( x[i]) , end='')
    for j in range (0 , i+1 ):
        print ('\t%0.2f ' %( y[i][j]) , end='')
    print ()
# obtaining the polynomial
t= symbols ('t')
f=[]
p=( t-x[n-1])/( x[1]-x[0])
f . append ( p )
for i in range (1 , n-1 ):
    f . append ( f[i-1]*( p+i )/( i+1 ) )
poly =y[n-1][0]
print ( poly )
for i in range ( n-1 ):
    poly = poly +y[n-1][i+1]*f[i]
    simp_poly = simplify ( poly )
print ('\n THE INTERPOLATING POLYNOMIAL IS\n') ;
pprint ( simp_poly )
# if you want to interpolate at some point the next session will help
inter = input ('Do you want to interpolate at a point (y/n)? ')
if inter =='y':
    a= float ( input ('enter the point ') )
    interpol = lambdify (t , simp_poly )
    result = interpol ( a )
    print ('\n The value of the function at ' ,a ,'is\n', result ) ;
```

Output :

```
 This will use Newton 's backword intepolation formula
Enter number of data points : 5
Enter data for x and y:
x[0]= 1
y[0]= 6
x[1]= 3
y[1]= 10
x[2]= 5
y[2]= 62
x[3]= 7
y[3]= 210
x[4]= 9
y[4]= 502
\ nBACKWARD DIFFERENCE TABLE

1.00    6.00
3.00    10.00   4.00
5.00    62.00   52.00   48.00
7.00    210.00  148.00  96.00   48.00
9.00    502.00  292.00  144.00  48.00   0.00
502.0

 THE INTERPOLATING POLYNOMIAL IS

     3        2
1.0·t  - 3.0·t  + 1.0·t + 7.0
Do you want to interpolate at a point (y/n)? y
enter the point 8

 The value of the function at  8.0 is
 335.0
```

7.2 Exercise:

1. Obtain the interpolating polynomial for the following data

x:	0	1	2	3
y:	1	2	1	10

Ans: $2x^3 - 7x^2 + 6x + 1$

2. Find the number of men getting wage Rs. 100 from the following table:

wage:	50	150	250	350
No. of men:	9	30	35	42

Ans: 23 men

3. Using Newton's backward interpolation method obtain y(160) for the following data

x:	100	150	200	250	300
y:	10	13	15	17	18

Ans: 13.42

4. Using Newtons forward interpolation polynomial and calculate $y(1)$ and $y(10)$.

x:	3	4	5	6	7	8	9
y:	4.8	8.4	14.5	23.6	36.2	52.8	73.9

Ans: 3.1 and 100

Viva Questions :

1. Newton- Gregory Forward interpolation formula can be used for both equally and unequally spaced intervals. True or False?
   Ans : False
2. When Newton's backward interpolation formula is used?
   Ans : The formula is used mainly to interpolate the values of y near the end of a set of tabular values and also for extrapolation the values of y a short distance ahead of y0
3. When do we apply Lagrange's interpolation?
   Ans : It is mainly used when the values are unevenly spaced.
4. When Newton's backward interpolation formula is used?
   Ans : The formula is used mainly to interpolate the values of y near the beginning of a set of tabular values and also for extrapolation the values of y a short distance before y0

LAB 8: Computation of area under the curve using Trapezoidal, Simpson's 1/3rd and simpson's 3/8th rule

8.1 Objectives :

Use python
1. to find area under the curve represented by a given function using Trapezoidal rule.
2. to find area under the curve represented by a given function using Simpson's 1/3rd rule.
3. to find area under the curve represented by a given function using Simpson's 3/8th rule.
4. to find the area below the curve when discrete points on the curve are given.

8.2 Trapezoidal Rule :

Example 1 : Evaluate $\int_0^5 \frac{1}{1+x^2}$ dx by using

Code :

```
# Definition of the function to integrate
def my_func ( x ):
    return 1 / ( 1 + x **2)
# Function to implement trapezoidal method
def trapezoidal ( x0 , xn , n ):
    h = ( xn - x0 ) / n                          # Calculating step size
# Finding sum
    integration = my_func ( x0 ) + my_func ( xn )    # Adding first and last terms
    for i in range (1 , n ):
        k = x0 + i * h                           # i-th step value
        integration = integration + 2 * my_func ( k )    # Adding areas of the trapezoids
# Proportioning sum of trapezoid areas
    integration = integration * h / 2
    return integration
# Input section
lower_limit = float ( input (" Enter lower limit of integration : ") )
upper_limit = float ( input (" Enter upper limit of integration : ") )
sub_interval = int ( input (" Enter number of sub intervals : ") )
# Call trapezoidal () method and get result
result = trapezoidal ( lower_limit , upper_limit , sub_interval )
# Print result
print (" Integration result by Trapezoidal method is: " , result)
```

Output :

```
Enter lower limit of integration: 0
Enter upper limit of integration: 5
Enter number of sub intervals: 10
Integration result by Trapezoidal method is:  1.3731040812301099
```

8.3 Simpson's $\left(\frac{1}{3}\right)^{rd}$ Rule

Example 2 : Evaluate $\int_0^5 \frac{1}{1+x^2} dx$ by using Simpson 's one - third rule.

Code :

```
# Definition of the function to integrate
def  my_func ( x ):
   return 1 / ( 1 + x **2)
# Function to implement the Simpson 's one - third rule
def simpson13 ( x0 , xn , n ):
   h = ( xn - x0 ) / n # calculating step size
# Finding sum
   integration = ( my_func ( x0 ) + my_func ( xn ) )
   k = x0
   for i in range (1 , n ):
      if i%2 == 0:
         integration = integration + 4 * my_func ( k )
      else :
         integration = integration + 2 * my_func ( k )
      k += h
# Finding final integration value
   integration = integration * h * ( 1/3 )
   return integration
# Input section
lower_limit = float ( input (" Enter lower limit of integration : ") )
upper_limit = float ( input (" Enter upper limit of integration : ") )
sub_interval = int ( input (" Enter number of sub intervals : ") )
# Call trapezoidal () method and get result
result = simpson13 ( lower_limit , upper_limit , sub_interval )
print (" Integration result by Simpson's 1/3 method is: %0.6f" % ( result ))
```

Output :

```
Enter lower limit of integration: 0
Enter upper limit of integration: 5
Enter number of sub intervals: 100
Integration result by Simpson's 1/3 method is: 1.404120
```

8.4 Simpson's $3/8$th rule

Example 2: Evaluate $\int_0^6 \frac{1}{1+x^2} dx$ using Simpson's $3/8$ th rule, taking 6 sub intervals.

```
def  simpsons_3_8_rule (f, a, b, n):
   h = (b - a) / n
```

```
    s = f(a) + f(b)
    for i in range (1, n, 3):
        s += 3 * f(a + i * h)
    for i in range (3, n-1, 3):
        s += 3 * f(a + i * h)
    for i in range (2, n-2, 3):
        s += 2 * f(a + i * h)
    return s * 3 * h / 8
def f(x):
    return 1/(1+x ** 2) # function here
a = 0                                   # lower limit
b = 6                                   # upper limit
n = 6                                   # number of sub intervals
result = simpsons_3_8_rule (f, a, b, n)
print ('%3.5f '% result)
```

Output:1.27631

8.5 Exercise :

1 Evaluate the integral $\int_0^1 \frac{x^2}{1+x^3} dx$ using Simpson's $\frac{1}{3}$ rule.
Ans: 0.23108

2 Use Simpson's $\frac{3}{8}$ rule to find $\int_0^{0.6} e^{-x^2} dx$ by taking seven ordinates,
Ans: 0.5351

3 Evaluate using trapezoidal rule $\int_0^{\pi} \sin^2 x dx$. Take $n = 6$.
Ans: $\pi/2$

4 A solid of revolution is formed by rotating about the x-axis, the area between the x-axis, the lines $x = 0$ and $x = 1$, and a curve through the points with the following co-ordinates:

x	y
0.00	1.0000
0.25	0.9896
0.50	0.9589
0.75	0.9089
1.00	0.8415

Estimate the volume of the solid formed using Simpson's $\frac{1}{3}$ rd rule.

Hint: Required volume is $\int_0^1 y^2 * \pi dx$. **[Ans: 2.8192]**

5. The velocity v(km/min) of a moped which starts from rest, is given at fixed intervals of time t(min) as follows:

t:	2	4	6	8	10	12	14	16	18	20
v:	10	18	25	29	32	20	11	5	2	0

Estimate approximately the distance covered in twenty minutes.

Answer for 5 :

We know that ds/dt = v. So to get distance (s) we have to integrate.

Here $h = 2.2, v_0 = 0, v_1 = 10, v_2 = 18, v_3 = 25$ etc.

Code :

```
# we shall use simpson's 1/3 rule directly to estimate
h=2
y= [0, 10 ,18 , 25 , 29 ,32 ,20 , 11 ,5 ,2 , 0]
result =(h/3)*((y[0]+y[10])+4*(y[1]+y[3]+y[5]+y[7]+y[9])+2*(y[2]+y[4]+y[6]+y[8]))
print('%3.5f '%result ,'km.')
```

Output :

309.33333 km.

Viva Questions :

1. What is the formula for evaluating the integral by using the trapezoidal rule?
   Ans : $\int_{x0}^{xn} f(x)dx = \frac{h}{2}[(y_0 + y_n) + 2(y_1 + y_2 + y_3 + y_5 + \cdots y_{n-1})]$
2. What is the formula for evaluating the integral by using the simpson's 1/3 rule?
   Ans : $\int_{x0}^{xn} f(x)dx = \frac{h}{3}[(y_0 + y_n) + 4(y_1 + y_3 + y_5 + \cdots) + 2(y_2 + y_4 + \cdots)]$
3. What is the formula for evaluating the integral by using the simpson's 3/8 rule?
   Ans : $\int_{x0}^{xn} f(x)dx = \frac{3h}{8}[(y_0 + y_n) + 3(y_1 + y_2 + y_4 + y_5 + \cdots) + 2(y_3 + y_6 + \cdots)]$
4. To use simpson's 1/3 rule, the number of subintervals should be even. True or False?
   Ans : True
5. To use simpson's 3/8 rule, the number of subintervals should be even. True or False?
   Ans : False

LAB 9: Solution of ODE of first order and first degree by Taylor's series and Modified Euler's method

9.1 Objectives:

Use python
1. to solve ODE by Taylor series method.
2. to solve ODE by Modified Euler method.
3. to trace the solution curves.

9.2 Taylor series method to solve ODE

Example 1: Solve: $\frac{dy}{dx} - 2y = 3e^x$ with y(0) = 0 using Taylor series method at x = 0.1(0.1)0.3.

Code :

```
import numpy as np
from numpy import array
def taylor(deriv ,x,y,xStop ,h):
    X = []
    Y = []
    X. append (x)
    Y. append (y)
    while x < xStop :    # Loop over integration steps
        D = deriv (x,y)    # Derivatives of y
        H = 1.0
        for j in range (3):          # Build Taylor series
            H = H*h/(j + 1)
            y = y + D[j]*H # H = h^j/j!
        x = x + h
        X. append (x)     # Append results to
        Y. append (y)     # lists X and Y
    return array (X),array (Y) # Convert lists into arrays
def deriv (x,y):
    D = np.zeros ((4,1))
    D[0] = [2*y[0] + 3*np.exp(x)]
    D[1] = [4*y[0]+ 9*np.exp(x)]
    D[2] = [8*y[0]+ 21*np.exp(x)]
    D[3] = [16*y[0]+ 45*np.exp(x)]
    return D
x = 0.0          # Initial value of x
xStop = 0.3      # last value
y = array ([0.0])        # Initial values of y
h = 0.1          # Step size
X,Y = taylor(deriv, x,y,xStop ,h)
print ("The required values are :at x= %0.2f, y=%0.5f, x=%0.2f, y=%0.5f, x = %0.2f,
y=%0.5f, x = %0.2f, y=%0.5f" %(X[0],Y[0],X[1],Y[1],X[2],Y[2],X[3],Y[3] ))
```

Output: The required values are :at x= 0.00, y=0.00000, x=0.10, y=0.34850,
x = 0.20, y=0.81079,x = 0.30, y=1.41590

Example 2: Solve $y' + 4y = x^2$ with initial conditions y(0) = 1 using Taylor series method at x =0.1, 0.2.

Code :

```
import numpy as np
from numpy import array
def taylor(deriv ,x,y,xStop ,h):
    X = []
    Y = []
    X. append (x)
    Y. append (y)
    while x < xStop : # Loop over integration steps
        D = deriv (x,y) # Derivatives of y
        H = 1.0
        for j in range (3): # Build Taylor series
            H = H*h/(j + 1)
            y = y + D[j]*H # H = h^j/j!
        x = x + h
        X. append (x) # Append results to
        Y. append (y) # lists X and Y
    return array (X),array (Y) # Convert lists into arrays
def deriv (x,y):
    D = np.zeros ((4,1))
    D[0] = [x ** 2-4*y[0]]
    D[1] = [2*x-4*x ** 2+16*y[0]]
    D[2] = [2-8*x+16*x ** 2-64*y[0]]
    D[3] = [-8+32*x-64*x ** 2+256*y[0]]
    return D
x = 0.0 # Initial value of x
xStop = 0.2 # last value
y = array ([1.0]) # Initial values of y
h = 0.1 # Step size
X,Y = taylor(deriv, x,y,xStop ,h)
print ("The required values are :at x= %0.2f , y=%0.5f , x=%0.2f , y=%0.5f, x = %0.2f ,
y=%0.5f"%(X[0],Y[0],X[1],Y[1],X[2],Y[2]))
```

Output :

The required values are : at x= 0.00 , y=1.00000 , x=0.10 , y=0.66967, x = 0.20 , y=0.45026

9.3 Euler's method to solve ODE:

To solve the ODE of the form $\frac{dy}{dx}$ =f(x, y) with initial conditions y(x0) = y0. The iterative formula is given by : y(x(i+1) = y(xi) + hf(xi, y(xi)).

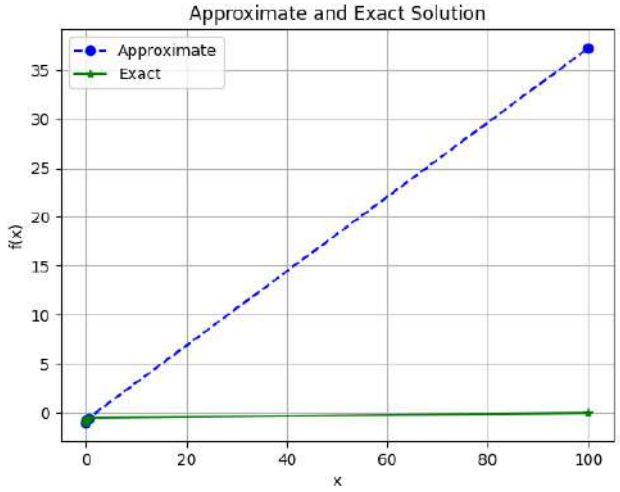
Example 3: Solve: $y' = e^{-x}$ with $y(0) = -1$ using Euler's method at $x = 0.2(0.2)0.6$.

Code :

```
Import numpy as np
Import matplotlib. pyplot as plt
# Define parameters
```

```
f = lambda x, y: np.exp(-x) # ODE
h = 0.2 # Step size
y0 = -1 # Initial Condition
n=3
# Explicit Euler Method
y[0] = y0
x[0]=0
for i in range (0, n):
        x[i+1]=x[i]+h
        y[i + 1] = y[i] + h*f(x[i], y[i])
print ("The required values are at x= %0.2f , y=%0.5f , x=%0.2f , y=%0.5f , x = %0.2f ,
y=%0.5f ,x = %0.2f , y=%0.5f"%(x[0],y[0],x[1],y[1],x[2],y[2],x[3], y[3]))
print ("\n\n")
plt .plot (x, y, 'bo --', label =' Approximate ')
plt .plot (x, -np.exp (-x), 'g*-', label ='Exact ')
plt .title (" Approximate and Exact Solution " )
plt .xlabel ('x')
plt .ylabel ('f(x)')
plt .grid ()
plt .legend (loc ='best ')
plt .show ()
```

Output: The required values are at x= 0.00, y=-1.00000, x=0.20, y=-0.80000, x = 0.40, y=-0.63625,x = 0.60, y=-0.50219



9.4 Modified Euler's method

The iterative formula is: $y_1^{(n+1)} = y_0 + \frac{h}{2}[f(x_0, y_0) + f\left(x_1, y_1^{(n)}\right]$, n = 0, 1, 2, 3, . . . ,
where, $y_1^{(n)}$ is the nth approximation to y_1.

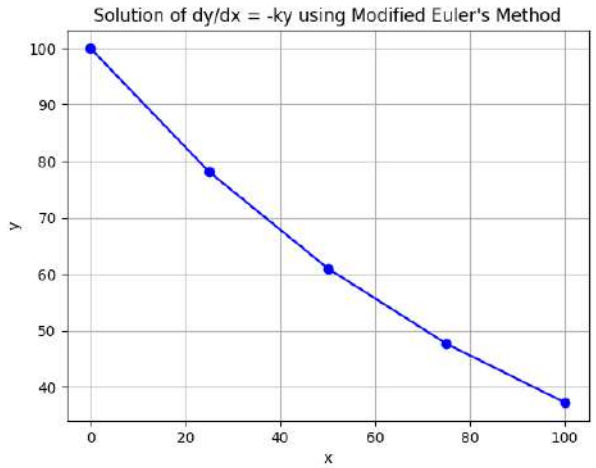
The first iteration will use Euler's method: $y_1^{(0)} = y_0 + hf(x_0, y_0)$.

Example 4: Solve $y' = -ky$ with $y(0) = 100$ using modified Euler's method at x = 100, by taking h = 25.

Code :

```
import numpy as np
import matplotlib. pyplot as plt
def modified_euler (f, x0 , y0 , h, n):
    x = np. zeros (n+1)
    y = np. zeros (n+1)
        x[0] = x0
        y[0] = y0
for i in range (n):
        x[i+1] = x[i] + h
         k1 = h * f(x[i], y[i])
        k2 = h * f(x[i+1], y[i] + k1)
        y[i+1] = y[i] + 0.5 * (k1 + k2)
return x, y
def f(x, y):
        return -0.01 * y          # ODE dy/dx = -ky
x0 = 0.0
y0 = 100 .0
h = 25
n = 4
x, y = modified_euler (f, x0 , y0 , h, n)
print ("The required value at x= %0.2f , y=%0.5f"%(x[4],y[4]))
print ("\n\n")
# Plotting the results
plt .plot (x, y, 'bo -')
plt .xlabel ('x')
plt .ylabel ('y')
plt .title ('Solution of dy/dx = -ky using Modified Euler \'s Method ')
plt .grid (True)
plt .show ()
```

Output: The required value at x= 100.00, y=37.25290



9.5 Exercise :

1. Find y(0.1) by Taylor Series exapnsion when $y' = x - y2$, $y(0) = 1$.
Ans: $y(0.1) = 0.9138$

2. Find y(0.2) by Taylor Series exapnsion when y′ = x2y − 1, y(0) = 1, h = 0.1.
Ans: y(0.2) = 0.80227

3. Evaluate by modified Euler's method: y′ = ln(x + y), y(0) = 2 at x = 0(0.2)0.8.
Ans: 2.0656, 2.1416, 2.2272, 2.3217

4. Solve by modified Euler's method: y′ = x + y, y(0) = 1, h = 0.1, x = 0(0.1)0.3.
Ans: 1.1105, 1.2432, 1.4004

Viva Questions :

1. In Euler's method: Given initial value problem y'=dy/dx=f(x, y) with y(x0) = y0, then approximation is given by __
   Ans : $y(x_1) = y_0 + f(x_0, y_0)$
2. y(x+h) = y(x) + h f(x,y) is referred as _____ method.
   Ans : Eulers method
3. In which of the numerical methods for solving differential equations, the calcul;ation of higher order serivatives become tedioucs.
   Ans : Taylor's series method
4. Modified Euler's formula is
   Ans : $\boldsymbol{y_1^{n+1} = y_0 + \frac{h}{2}(f(x_0, y_0) + f(x_1, y_1^n))}$
5. Taylors series formula is
   Ans : $\boldsymbol{y(x) = y_0 + (x - x_0)y_0' + \frac{(x-x_0)y_0''}{2!} + \cdots}$

LAB 10: Solution of ODE of first order and first degree by Runge-Kutta 4th order method and Milne's predictor and corrector method

10.1 Objectives:

1. To write a python program to solve first order differential equation using 4th order Runge Kutta method.
2. To write a python program to solve first order differential equation using Milne's predictor and corrector method.

10.2 Runge-Kutta method :

Example 1 : Apply the Runge Kutta method to find the solution of dy/dx = 1 + (y/x) at y(2) taking h = 0.2. Given that y(1) = 2.

Code :

```
from sympy import *
import numpy as np
def RungeKutta (g,x0 ,h,y0 ,xn):
    x,y= symbols ('x,y')
    f= lambdify ([x,y],g)
    xt=x0+h
    Y=[y0]
    while xt<=xn:
        k1=h*f(x0 ,y0)
        k2=h*f(x0+h/2, y0+k1/2)
        k3=h*f(x0+h/2, y0+k2/2)
        k4=h*f(x0+h, y0+k3)
        y1=y0+(1/6)*(k1+2*k2+2*k3+k4)
        Y. append (y1)
            # print ('y(%3.3f '%xt ,') is %3.3f '%y1)
        x0=xt
        y0=y1
        xt=xt+h
    return np. round (Y,2)
RungeKutta ('1+(y/x)',1,0.2,2,2)
```

Output: array([2. , 2.62, 3.27, 3.95, 4.66, 5.39])

10.3 Milne's predictor and corrector method

Example 2 : Apply Milne's predictor and corrector method to solve dy/dx = x2 + (y/2) at y(1.4). Given that y(1)=2, y(1.1)=2.2156, y(1.2)=2.4649, y(1.3)=2.7514. Use corrector formula thrice.

Code :

```
# Milne 's method to solve first order DE
# Use corrector formula thrice
x0=1
y0=2
y1=2.2156
y2=2.4649
y3=2.7514
h=0.1
x1=x0+h
x2=x1+h
x3=x2+h
x4=x3+h
def f(x,y):
   return x ** 2+(y/2)
y10 =f(x0 , y0)
y11 =f(x1 ,y1)
y12 =f(x2 ,y2)
y13 =f(x3 ,y3)
y4p =y0+(4*h/3)*(2*y11-y12+2*y13)
print ('predicted value of y4 is %3.3f '%y4p)
y14 =f(x4 ,y4p );
for i in range (1,4):
   y4=y2+(h/3)*(y14 +4*y13 +y12 );
   print ('corrected value of y4 after \t iteration %d is \t %3.5f\t '%(i,y4))
   y14=f(x4 ,y4);
```

Output:

```
predicted value of y4 is 3.079
corrected value of y4 after iteration 1 is 3.07940
corrected value of y4 after iteration 2 is 3.07940
corrected value of y4 after iteration 3 is 3.07940
```

In the next program, function will take all the inputs from the user and display the answer.

Example 2 : Apply Milne's predictor and corrector method to solve $\frac{dy}{dx} = x^2 + \left(\frac{y}{2}\right)$at y(1.4). Given that y(1)=2, y(1.1)=2.2156, y(1.2)=2.4649, y(1.3)=2.7514. Use corrector formula thrice.

```
from sympy import *
def Milne (g,x0 ,h,y0 ,y1 ,y2 ,y3):
        x,y= symbols ('x,y')
        f= lambdify ([x,y],g)
        x1=x0+h
        x2=x1+h
        x3=x2+h
        x4=x3+h
         y10=f(x0 , y0)
        y11=f(x1 ,y1)
        y12=f(x2 ,y2)
        y13=f(x3 ,y3)
```

```
        y4 p=y0+(4*h/3)*(2*y11-y12+2*y13)
        print ('predicted value of y4 ',y4p )
        y14=f(x4 ,y4p)
        for i in range (1,4):
                y4=y2+(h/3)*(y14 +4*y13 +y12 )
                print ('corrected value of y4 , iteration %d '%i,y4)
                y14=f(x4 ,y4)
  Milne ('x**2+y/2',1,0.1,2,2.2156 ,2.4649 ,2.7514 )
```

Output:predicted value of y4 3.0792733333333335
corrected value of y4 , iteration 1 3.0793962222222224
corrected value of y4 , iteration 2 3.079398270370371
corrected value of y4 , iteration 3 3.079398304506173

Example 3 : Apply Milne's predictor and corrector method to solve $\frac{dy}{dx} = x - y^2$, y(0)=2 obtain y(0.8). Take h=0.2. Use Runge-Kutta method to calculate required initial values.

```
Y= RungeKutta ('x-y**2',0,0.2,0,0.8)
print ('y values from Runge -Kutta method :',Y)
Milne ('x-y**2',0,0.2,Y[0],Y[1],Y[2],Y[3])
```

y values from Runge -Kutta method: [0. 0.02 0.08 0.18 0.3]
predicted value of y4 0.3042133333333334
corrected value of y4 , iteration 1 0.3047636165214815
corrected value of y4 , iteration 2 0.3047412758696499
corrected value of y4 , iteration 3 0.3047421836520892

10.4 Exercise:

1. Find y(0.1) by RungeKutta method when y′ = x − y2, y(0) = 1.
Ans: y(0.1) = 0.91379

2. Evaluate by RungeKuttamethod :y′ = log(x + y), y(0) = 2 at x = 0(0.2)0.8.
Ans: 2.155, 2.3418, 2.557, 2.801

3. Solve by Milnes method: y′ = x + y, y(0)=1, h=0.1, Calculate y(0.4) . Calculate required initial values from RungeKutta method.
Ans: 1.583649219

Viva Questions :

1. How many prior values are required to predict the next value in Miline's method?
   Ans : 4
2. Milnes predictor formula is ---
   Ans : $y_{4,p} = y_0 + \frac{4h}{3}\left[2y_1^{'} - y_2^{'} + 2y_3^{'}\right]$
3. Milnes corrector formula is -----
   Ans : $y_{4,c} = y_2 + \frac{h}{3}\left[y_2^{'} + 4y_3^{'} + y_4^{'}\right]$
4. In the R.K method of order 4 the value of k_1 is ----
   Ans : $hf(x_0, y_0)$