Practical Test
CS2883 Object Oriented Programming for Mechanical Engineers

Duration: 2h 30 min
Total marks Allocated:

Make necessary design decisions to add value to the output rather than devaluing the output.

(Q1): Simulation of Stress Distribution in a Point-Loaded Cantilever Beam (**25 Marks**)

Figure Q1 illustrates a cantilever beam subjected to a point load applied at its free end. The beam's cross-section can be square, rectangular, or circular, as shown on the right side of the diagram, with dimensions labeled as *h* (height), *w* (width), and *r* (radius). The beam length is L, and the applied load is P. The position along the beam is denoted by the distance *x*, measured from the fixed end.

The bending axis's second moment of inertia (I) for the cross-section is described by Equations 1 and 2, applicable to square/rectangular and circular geometries, respectively. The bending moment at a distance *x* from the fixed support (Mx) is determined using Equation 3, while Equation 4 calculates the corresponding maximum bending stress (σ_x) at that position.

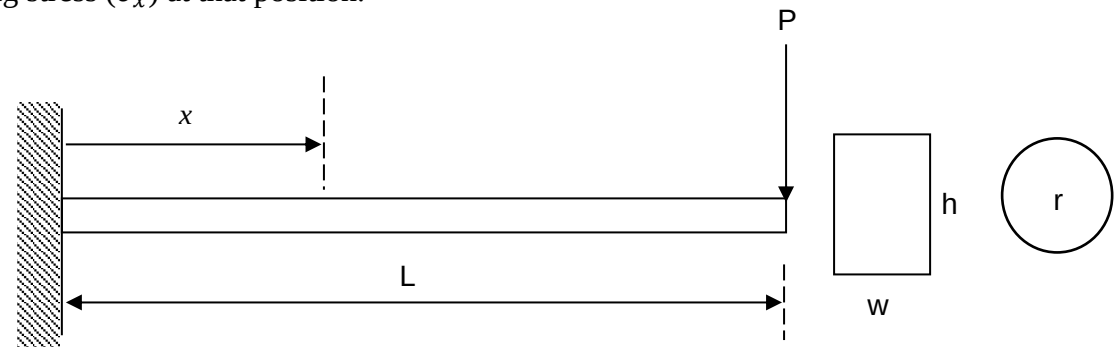

Fig. Q1

$$I = \frac{wh^3}{12} \rightarrow (1)$$

$$I = \frac{\pi r^4}{4} \rightarrow (2)$$

$$M_x = P(L - x) \rightarrow (3)$$

$$\sigma_x = \frac{M_x y}{I} \rightarrow (4)$$

where y = h/2 for rectangle and y = r for the circle

To facilitate engineering design, it is essential to compute the distribution of maximum bending stress along the entire length of the beam at specified intervals. Develop a program that adheres to the following requirements.

a) **Implementation of Class Interface**: Utilize the provided class interface file (*Bending.h*), which includes detailed functional specifications and descriptions of the data members, to implement the required functionality.

b) **Development of the Main Application**: Create the main application with the following capabilities to perform the stress analysis. Refer to the output layout shown in the accompanying figure for clarity.
   - **Input Beam Cross-Section**: Prompt the user to select the cross-sectional shape of the beam.
   - **Input Beam Dimensions**: Collect the dimensions of the beam based on the selected cross-sectional shape.
   - **Input Loading and Analysis Parameters**: Gather the applied load and analysis details from the user. Specifically, request the number of points for stress calculation. For example, if the user specifies 10 points, divide the total beam length into 10 equal segments and compute the stress at each point, from x=0x = 0x=0 to the beam's total length.

c) **Program Output**: Ensure the program's output matches the layout depicted in the figure provided.

```
Bending Stress Calculation
         Circle (c)
         Square (s)

Select the shape (c or s)       s

Provide the beam dimensions in the given units

Length (m)      8

Width (cm)      5

Height (cm)     8

Provide the loading and analysis details

Point Load (kN)         15

Number of points required       10

Stress distribution

        Distance        Stress (MPa)
        0               2250
        0.8             2025
        1.6             1800
        2.4             1575
        3.2             1350
        4               1125
        4.8             900
        5.6             675
        6.4             450
        7.2             225
        8               2.498e-13
```

(Q2): Matrix manipulation **(15 Marks)**

The supplied class interface and object file includes a matrix manipulation class named **MatrixBase**, designed to handle 3x3 matrices. This class provides functionality for assigning values to the matrix using the *SetMatrix* fuction and displaying the matrix on the screen through the *Show* function. Additionally, the default constructor initializes all elements of the matrix to zero upon object creation.

Create a class `Matrix` inherited from `MatrixBase` which has the following capabilities:

a. All elements must be initialized to zero when a matrix is created.

b. Overload the addition (+), subtraction (-) and add and assignment operator (+=). Further, overload the multiplication operator (*) such that scalar multiplication of matrices can be performed whether the scalar is the right or the left operand.

Write a program with the following features. Please refer to the screenshot of the program output given below.

a. The user demines the number of matrices to be created. Thus, prompt it from the user.
b. Get the element values for each of the matrices the user wants to create. (r[1]c[1] refers to the element on the first row and first column)
c. Once elements are entered, printed each of the matrix on screen.
d. Add all matrices together and print the results on the screen.
e. Perform the following matrix operations over the first two matrices, namely A and B, and print the results on the screen.
   (i) 2A + 3B

(ii) 5B – 3A

```
What is the number of matrices required?        2

Provide the matrix elements


Elements of Matrix 1
Element r[0]c[0]        1
Element r[0]c[1]        5
Element r[0]c[2]        6
Element r[1]c[0]        8
Element r[1]c[1]        4
Element r[1]c[2]        3
Element r[2]c[0]        12
Element r[2]c[1]        18
Element r[2]c[2]        12


Elements of Matrix 2
Element r[0]c[0]        25
Element r[0]c[1]        10
Element r[0]c[2]        -8
Element r[1]c[0]        12
Element r[1]c[1]        -3
Element r[1]c[2]        -15
Element r[2]c[0]        16
Element r[2]c[1]        14
Element r[2]c[2]        0

Matrix 1
        1       5       6
        8       4       3
        12      18      12

Matrix 2
        25      10      -8
        12      -3      -15
        16      14      0


Sum of all matrices
        26      15      -2
        20      1       -12
        28      32      12
Matrices 2A + 3B
        77      40      -12
        52      -1      -39
        72      78      24

Matrices 5B - 3A
        122     35      -58
        36      -27     -84
        44      16      -36
```