

COURSE TEMPLATE

1.	Department/Centre proposing the course	CSE
2.	Course Title (<i>< 45 characters</i>)	INTRODUCTION TO COMPUTING
3.	L-T-P structure	0-0-4
4.	Credits	2
5.	Course number	CSP100
6.	Status (<i>category for program</i>)	All UG Programmes

7.	Pre-requisites (<i>course no./title</i>)	None
----	--	------

8.	Status vis-à-vis other courses (<i>give course number/title</i>)	
8.1	Overlap with any UG/PG course of the Dept./Centre	None
8.2	Overlap with any UG/PG course of other Dept./Centre	None
8.3	Supercedes any existing course	None

9.	Not allowed for (<i>indicate program names</i>)	
----	---	--

10.	Frequency of offering	☒ Every sem ☐ 1 st sem ☐ 2 nd sem ☐ Either sem
-----	------------------------------	---

11.	Faculty who will teach the course Multiple faculty members may be assigned. This is a first year course.	
-----	--	--

12.	Will the course require any visiting faculty?	No
-----	--	----

13.	Course objective (<i>about 50 words</i>): Introduction to computing infrastructure; responsible and effective usage. Use of high level languages for problem solving using computational tools: • Map scientific problems into computational frameworks. • Making effective use of computational methods in the student's chosen field.	
-----	---	--

14.	Course contents (<i>about 100 words</i>) (<i>Include laboratory/design activities</i>): Introduction to computing systems and software. Responsible and effective usage practices. Concept of an algorithm; termination and correctness. Algorithms to programs: specification, top-down development and stepwise refinement. • Learning a language for expressing computations	
-----	--	--

Dept.: COMP. SC. ENGG.

04 FEB 2013

Ref. No.: 20...13...BCSE...308

- | | |
|--|--|
| | <ul style="list-style-type: none">• The process of writing and debugging a program;• The process of moving from a problem statement to a computational formulation of a method for solving the problem;• Using simulations to model problems without easy closed form solutions• Using computational tools to help model and understand data• Applying computational modes of thought to frame problems, and to guide the process of deducing information in a computational manner. |
|--|--|

15. Lecture Outline (with topics and number of lectures)

Module no.	Topic	No. of hours
1		
2		
3		
4		
5		
6		
7		
8	I	
9		
10		
11		
12		
COURSE TOTAL (14 times 'L')		

16. Brief description of tutorial activities

NA

17. Brief description of laboratory activities

Module no.	Experiment description	No. of hours
1	Getting started: Introduction to Computers and their usage in a networked shared environment. Secure use of user accounts. Searching for information on the WWW efficiently. Navigating through a directory system, shell commands. Creating a program and executing it.	4
2	Formulating a problem as a computational problem: representing data using standard supported types. Transforming input to output using basic library functions	4
3	Functional transformations of data. Decomposing a problem into more elementary problems. Recursive functions for dealing with similar sub-problems. Relationship of recursion to inductive reasoning	4
4	Elements of the imperative model of programming: storing values in variables. Sequential and conditional control structures.	4
5	Tail recursion and iteration. Bounded iteration (for)	4
6	Step-wise refinement. Procedural abstraction.	4
7	Input and output. Interfacing with other resources on the system.	4
8	Data models for more complex data. Interfaces.	4
9	Iterative computation: Newton-Raphson. Ensuring termination	4
10	Modelling matrices and other mathematical structures in a variety of programming applications	20
COURSE TOTAL (14 times 'P')		56

18. Suggested texts and reference materials

STYLE: Author name and initials, Title, Edition, Publisher, Year.

1. How to Solve It: A New Aspect of Mathematical Method (Princeton Science Library) G.

- Polya
2. Online course material and video material.

19. Resources required for the course (*itemized & student access requirements, if any*)

19.1	Software	Matlab and Language Interpreters
19.2	Hardware	Computer Terminals for All Students
19.3	Teaching aides (videos, etc.)	
19.4	Laboratory	Computer Centre/First Year Teaching Lab
19.5	Equipment	
19.6	Classroom infrastructure	Classrooms for students
19.7	Site visits	None

20. Design content of the course (*Percent of student time with examples, if possible*)

20.1	Design-type problems	20%
20.2	Open-ended problems	
20.3	Project-type activity	
20.4	Open-ended laboratory work	
20.5	Others (please specify)	

Date: Feb 4, 2013

(Signature of the Head of the Department)