COMPUTER SCIENCE AND ENGINEERING

| **CST424** | **PROGRAMMING**  **PARADIGMS** | **CATEGORY** | **L** | **T** | **P** | **CREDIT** | **YEAR OF**  **INTRODUCTION** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **PEC** | **2** | **1** | **0** | **3** | **2019** |

**Preamble**: The course provides the learners a clear understanding of the main constructs of contemporary programming languages and the various systems of ideas that have been used to guide the design of programming languages. This course covers the concepts of Names, Bindings & Scope, Statement-Level Control Structures, Sub Programs, Support for Object Oriented Programming, Exception Handling, Concurrency Control, Functional Programming and Logic Programming. This course helps the learners to equip with the knowledge necessary for the critical evaluation of existing and upcoming programming languages. It also enables the learner to choose the most appropriate language for a given programming task, apply that language's approach to structure or organize the code, classify programming languages based on their features and to design new generation languages.

**Prerequisite:** Sound knowledge in Programming in C and Object-Oriented Programming. **Mapping of course outcomes with program outcomes**

| **CO1** | Explain the criteria for evaluating programming languages and compare Imperative, Functional and Logic programming languages (**Cognitive Knowledge Level: Understand**) |
| --- | --- |
| **CO2** | Illustrate the characteristics of data types and variables (**Cognitive Knowledge Level: Apply**) |
| **CO3** | Comprehend how control flow structures and subprograms help in developing the structure of a program to solve a computational problem (**Cognitive Knowledge Level: Apply)** |
| **CO4** | Explain the characteristics of Object-Oriented Programming Languages (**Cognitive Knowledge Level: Understand)** |
| **CO5** | Compare concurrency constructs in different programming languages (**Cognitive Knowledge Level: Understand)** |

COMPUTER SCIENCE AND ENGINEERING

**Mapping of course outcomes with program outcomes**

|  | **PO1** | **PO2** | **PO3** | **PO4** | **PO5** | **PO6** | **PO7** | **PO8** | **PO9** | **PO10** | **PO11** | **PO12** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **CO1** |  |  |  |  |  |  |  |  |  |  |  |  |
| **CO2** |  |  |  |  |  |  |  |  |  |  |  |  |
| **CO3** |  |  |  |  |  |  |  |  |  |  |  |  |
| **CO4** |  |  |  |  |  |  |  |  |  |  |  |  |
| **CO5** |  |  |  |  |  |  |  |  |  |  |  |  |

| **Abstract POs defined by National Board of Accreditation** | | | |
| --- | --- | --- | --- |
| **PO#** | **Broad PO** | **PO#** | **Broad PO** |
| **PO1** | Engineering Knowledge | **PO7** | Environment and Sustainability |
| **PO2** | Problem Analysis | **PO8** | Ethics |
| **PO3** | Design/Development of solutions | **PO9** | Individual and team work |
| **PO4** | Conduct investigations of  complex problems | **PO10** | Communication |
| **PO5** | Modern tool usage | **PO11** | Project Management and Finance |
| **PO6** | The Engineer and Society | **PO12** | Life long learning |

**Assessment Pattern**

| **Bloom’s**  **Category** | **Continuous Assessment Tests** | | **End Semester Examination Marks (%)** |
| --- | --- | --- | --- |
| **Test 1 (%)** | **Test 2 (%)** |
| Remember | **30** | **30** | **30** |
| Understand | **40** | **40** | **40** |

COMPUTER SCIENCE AND ENGINEERING

| Apply | **30** | **30** | **30** |
| --- | --- | --- | --- |
| Analyze |  |  |  |
| Evaluate |  |  |  |
| Create |  |  |  |

**Mark Distribution**

| **Total Marks** | **CIE Marks** | **ESE Marks** | **ESE Duration** |
| --- | --- | --- | --- |
| **150** | **50** | **100** | **3** |

**Continuous Internal Evaluation Pattern:**

Attendance **10 marks** Continuous Assessment Tests (Average of Internal Tests 1 & 2) **25 marks** Continuous Assessment Assignment **15 marks**

**Internal Examination Pattern**

Each of the two internal examinations has to be conducted out of 50 marks. First series test shall be preferably conducted after completing the first half of the syllabus and the second series test shall be preferably conducted after completing the remaining part of the syllabus. There will be two parts: Part A and Part B. Part A contains 5 questions (preferably, 2 questions each from the two completed modules and 1 question from the partly completed module), having 3 marks for each question adding up to 15 marks for part A. Students should answer all questions from Part A. Part B contains 7 questions (preferably, 3 questions each from the completed two modules and 1 question from the partly completed module), each with 7 marks. Out of the 7 questions, a student should answer any 5.

**End Semester Examination Pattern:**

There will be two parts; Part A and Part B. Part A contains 10 questions with 2 questions from each module, having 3 marks for each question. Students should answer all questions. Part B contains 2 full questions from each module of which student should answer any one. Each question can have maximum 2 sub-divisions and carries 14 marks.

**Course Level Assessment Questions Course Outcome1 (CO1):**

COMPUTER SCIENCE AND ENGINEERING

1. Compare any three programming languages based on the language evaluation criteria. Prepare a list of characteristics that affect the language evaluation criteria.

2. Identify the advantages and disadvantages of imperative, functional and logic programming languages.

**Course Outcome 2 (CO2):**

1. Two most important design issues that are specific to character string types are (1) whether a string is simply a special kind of character array or a primitive type. (2) whether strings have static or dynamic length.

Identify the implementations options for the above two cases.

2. Consider the following records of a particular language. Let the size of each char variable be 1 byte, int be 4 bytes and and Boolean be 1 bit.

Struct Student

{

int id;

char name[2];

int age;

boolean scholarship;

}

Draw and comment on the possible memory layouts for the record for a 32-bit aligned machine

**Course Outcome 3(CO3):**

1. Explain three situations where a combined counting and logical looping statement is needed.

2. Describe the ways that aliases can occur with pass-by-reference parameters. 3. Identify the two fundamental design considerations for parameter-passing methods.

4. What will be the output of the given program segment if it uses the following parameter passing mechanisms:

a) call by reference

b) call by value

**x : integer – – global**

**procedure foo(y : integer)**

**y := 3**

**print x**

**. . .**

**x := 2**

**foo(x)**

**print x**

**Course Outcome 4 (CO4):**

COMPUTER SCIENCE AND ENGINEERING

1. Describe the role of a virtual method table in implementing dynamic method binding. 2. Identify the merits and demerits of inheritance.

**Course Outcome 5 (CO5):**

**1.** Evaluate the use of semaphores and monitors for providing competition synchronization and cooperation synchronization.

**Syllabus**

**Module – 1**

Introduction – Role of Programming Languages, Programming Domains, Language Evaluation Criteria, Influence on Language Design, Language Design Trade-offs, Implementation Methods. Names, Bindings & Scope – Names, Variables, Concept of Binding, Scope and Lifetime, Referencing Environments.

**Module - 2**

Data Types – Primitive Data Types, Character String Types, User-Defined Ordinal Types, Array Types, Record Types, List Types, Pointer & Reference Types, Type Checking, Strong Typing, Type Equivalence. Expressions – Arithmetic Expressions, Overloaded Operators, Type Conversions, Relational and Boolean Expressions, Short-Circuit Evaluation. Assignment - Assignment Statements, Mixed-mode Assignment.

**Module** - 3

Statement-Level Control Structures – Selection Statements, Iterative Statements, Unconditional Branching, Guarded Commands. Subprograms – Design Issues of Subprograms, Local Referencing Environments, Parameter Passing Methods, Subprograms as Parameters, Overloaded Subprograms, Closures, Co-routines

**Module** - 4

Support for Object Oriented Programming – Inheritance, Dynamic Binding, Design Issues for Object Oriented Languages, Support for Object Oriented Programming in C++, Implementation of Object-oriented Constructs. Exception Handling – Basic Concepts, Design Issues.

**Module** - 5

Concurrency – Subprogram Level Concurrency, Semaphores, Monitors, Message Passing. Functional Programming Languages – Introduction to LISP and Scheme, Comparison of

COMPUTER SCIENCE AND ENGINEERING

Functional and Imperative Languages. Logic Programming Languages – Basic Elements of Prolog, Applications of Logic Programming.

**Text Books**

1. Robert W Sebesta, Concepts of Programming Languages, 10th Edition, Pearson.

2. Scott M L, Programming Language Pragmatics, 3rd Edition, Morgan Kauffman Publishers.

**ReferenceBooks**

1. Kenneth C. Louden, Programming Languages: Principles and Practice, 2nd Edition, Cengage Learning.

2. Tucker A. B. and R. E. Noonan, Programming Languages: Principles and Paradigms, 2nd Edition. –TMH.

3. Ravi Sethi, Programming Languages: Concepts & Constructs, 2nd Edition., Pearson Education.

4. David A. Watt, Programming Language Design Concepts, Wiley Dreamtech.

**Model Question Paper**

**QP CODE:**

**Reg No: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ PAGES : 4 APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY**

**EIGHTH SEMESTER B.TECH DEGREE EXAMINATION, MONTH & YEAR Course Code: CST424**

**Course Name: Programming Paradigms**

**Max. Marks : 100 Duration: 3 Hours PART A**

**Answer All Questions. Each Question Carries 3 Marks**

**1.** Differentiate between readability and writability.

**2.** Define binding and binding time.

**3.** What are the advantages of user-defined enumeration types?

**4.** Define narrowing and widening conversions.

**5.** Why for statement in C language is more flexible than that of older languages?

COMPUTER SCIENCE AND ENGINEERING

**6.** What are the advantages and disadvantages of dynamic local variables in subprograms?

**7.** Illustrate the concept of dynamic method binding with an example.

**8.** Is it mandatory to use constructors in object-oriented languages? Justify your answer. **9.** What are the applications of logic programming languages?

**10.**Explain the working of let and let-rec constructs in Scheme.

**(10x3=30)**

**Part B**

**(Answer any one question from each module. Each question carries 14 Marks) 11.**(a) Explain different criteria used for evaluating languages. **(7)**

(b) Consider the following pseudocode:

**(7)**

**x : integer := 3**

**y : integer := 4**

**procedure add**

**x := x + y**

**procedure second(P : procedure)**

**x : integer := 5**

**P()**

**procedure first**

**y : integer := 6**

**second(add)**

**first()**

**write integer(x)**

(a) What does this program print if the language uses static scoping? Give reasons.

(b) What does it print if the language uses dynamic scoping? Give reasons. **OR**

**12.**(a) With respect to storage binding, explain the meanings, purposes, advantages and

**(7)**

disadvantages of four categories of scalar variables.

(b) What is meant by referencing environment of a statement? Show the **(7)**

COMPUTER SCIENCE AND ENGINEERING

referencing environment at the indicated program points (1), (2), (3) & (4) for the following program segment. Assume that the programming language is statically scoped.

**program example;**

**var a, b : integer;**

**procedure sub1;**

**var x, y: integer;**

**begin { sub1 }**

**……….. (1)**

**end { sub1 }**

**procedure sub2;**

**var x : integer;**

**……..**

**procedure sub3;**

**var x: integer;**

**begin { sub3 }**

**……….. (2)**

**end { sub3 }**

**begin { sub2 }**

**……….. (3)**

**end { sub2}**

**begin {example}**

**……….. (4)**

**end {example }**

**13.**(a) Explain any two issues associated with the pointer data types and also indicate

**(7)**

how dangling pointer problem can be solved.

(b) Describe the lazy and eager approaches for reclaiming garbage. **(7) OR**

**14.**(a) What is meant by side effect and illustrate the advantages of referential

**(8)**

transparency?

(b) Explain the terms: compound assignment operator, coercion and short circuit

**(6)**

evaluation.

COMPUTER SCIENCE AND ENGINEERING

**15.**(a) Illustrate the different categories of iteration control statements. **(8)**

(b) Explain the techniques used for identifying the correct referencing environment

**(6)**

for a subprogram that was sent as a parameter.

**OR**

**16.**(a) Describe the implementation models of Parameter passing. **(10)** (b) Differentiate coroutines from conventional subprograms. **(4)**

**17.**(a) What is meant by an exception handler? Explain how exceptions are handled in

**(7)**

object-oriented languages.

(b) Describe the design issues in object-oriented languages. **(7) OR**

**18.**(a) Illustrate how a virtual method table can be used for implementing dynamic

**(7)**

method binding.

(b) Explain the different categories, merits and demerits of inheritance. **(7) 19.**(a) Compare functional and imperative programming languages. **(7)** (b) Explain the role of monitors in concurrency. **(7) OR**

**20.**(a) Explain the searching strategies used in Prolog. Why backward chaining is preferred over forward chaining in Prolog?

(b) **(let ((a 6)**

**(b 8)**

**(square (lambda (x) (\* x x)))**

**(plus +))**

**(sqrt (plus (square a) (square b))))**

Write the output of the above code? Explain how let and lambda construct works?

**(10) (4)**

COMPUTER SCIENCE AND ENGINEERING

**Teaching Plan**

| **No** | **Contents** | **No. of**  **Lecture**  **Hours**  **(36 hrs.)** |
| --- | --- | --- |
| **Module-1 (7 hours)** | | |
| 1.1 | Introduction: Reasons for studying Concepts of programming languages, Programming Domains | 1 hour |
| 1.2 | Language Evaluation Criteria | 1 hour |
| 1.3 | Influence on Language Design, Language Design Trade-offs | 1 hour |
| 1.4 | Implementation Methods | 1 hour |
| 1.5 | Names, Variables | 1 hour |
| 1.6 | Concept of Binding | 1 hour |
| 1.7 | Scope and Lifetime, Referencing Environments | 1 hour |
| **Module-2 (7 hours)** | | |
| 2.1 | Primitive Data Types, Character String Types | 1 hour |
| 2.2 | User-Defined Ordinal Types, Array Types | 1 hour |
| 2.3 | Record Types, List Types, Pointer and Reference Types | 1 hour |
| 2.4 | Implementation of pointer and reference types, Type Checking, Strong Typing, Type Equivalence | 1 hour |
| 2.5 | Expressions and Assignment Statements, Arithmetic Expressions | 1 hour |
| 2.6 | Overloaded Operators, Type Conversions | 1 hour |
| 2.7 | Relational and Boolean Expressions, Short-Circuit Evaluation, Assignment Statements, Mixed-mode Assignment | 1 hour |
| **Module-3 (8 hours)** | | |
| 3.1 | Selection Statements, Iterative Statements | 1 hour |
| 3.2 | Unconditional Branching | 1 hour |

COMPUTER SCIENCE AND ENGINEERING

| 3.3 | Guarded Commands | 1 hour |
| --- | --- | --- |
| 3.4 | Subprograms: Design Issues of Subprograms | 1 hour |
| 3.5 | Local Referencing Environments | 1 hour |
| 3.6 | Parameter Passing Methods | 1 hour |
| 3.7 | Subprograms as Parameters, Overloaded Subprograms | 1 hour |
| 3.8 | Closures, Co-routines | 1 hour |
| **Module-4 (7 hours)** | | |
| 4.1 | Inheritance | 1 hour |
| 4.2 | Dynamic Binding | 1 hour |
| 4.3 | Design Issues for Object Oriented Languages | 1 hour |
| 4.4 | Support for Object Oriented Programming in C++ | 1 hour |
| 4.5 | Implementation of Object-Oriented Constructs | 1 hour |
| 4.6 | Exception Handling – Basic Concepts | 1 hour |
| 4.7 | Exception Handling - Design Issues | 1 hour |
| **Module-5 (7 hours)** | | |
| 5.1 | Subprogram Level Concurrency | 1 hour |
| 5.2 | Semaphores, Monitors | 1 hour |
| 5.3 | Message Passing | 1 hour |
| 5.4 | Introduction to LISP and Scheme | 1 hour |
| 5.5 | Comparison of Functional and Imperative Languages | 1 hour |
| 5.6 | Basic Elements of Prolog | 1 hour |
| 5.7 | Applications of Logic Programming | 1 hour |