# FR. Conceicao Rodrigues College Of Engineering

Father Agnel Ashram, Bandstand, Bandra-west, Mumbai-50
**Department of Computer Engineering**

**T.E. (CE-B) (Semester VI)**

**Course Outcomes & Assessment Plan**

**Subject:** System Programming Compiler Construction

**Subject code:** (CSC601)

**Teacher-in-charge: Prof. Sangeeta Parshionikar**

**Academic Term:  January – April 2023**

**Syllabus:**

**1. Introduction to System Software**

Concept of System Software, Goals of system software's, system program and system programming, Introduction to various system programs such as Assembler, Macro processor, Loader, Linker, Compiler, Interpreter, Device Drivers, Operating system, Editors, Debuggers

**2. Assemblers**

Elements of Assembly Language programming, Assembly scheme, pass structure of assembler, Assembler Design: Two pass assembler Design and single pass Assembler Design for Hypothetical / X86 family processor, data structures used.

**3.Macros & Macro processors**

Introduction, Macro definition and call, Features of Macro facility: Simple, parameterized, conditional and nested. Design of single pass macro processor, data structures used

**4.Loaders and Linkers**

Introduction, functions of loaders, Relocation and Linking concept, Different loading schemes: Relocating loader, Direct Linking Loader, Dynamic linking and loading.

**5. Compilers: Analysis Phase**

Introduction to compilers, Phases of compilers: **Lexical Analysis**- Role of Finite State Automata in Lexical Analysis, Design of Lexical analyser, data structures used . **Syntax Analysis**- Role of Context Free Grammar in Syntax analysis, Types of Parsers: Top down parser- LL(1), Bottom up parser- Operator precedence parser, SLR **Semantic Analysis**, Syntax directed definitions.

**6. Compilers: Synthesis Phase**

**Intermediate Code Generation**: Types of Intermediate codes: Syntax tree, Postfix notation, Three address codes: Triples and Quadruples. **Code Optimization**: Need and sources of optimization, Code optimization techniques: Machine Dependent and Machine Independent. **Code Generation**: Issues in the design of code generator, code generation algorithm. Basic block and flow graph.

**Text Books:**
1. D. M Dhamdhere: Systems programming, Tata McGraw Hill
**2. A. V. Aho, R. Shethi, Monica Lam , J.D. Ulman : Compilers Principles, Techniques and Tools ,** Pearson Education , Second Edition.
**3. J. J. Donovan: Systems Programming Tata McGraw Hill Publishing Company**

**Reference Books:**
1. Lex and Yacc , 2$^{nd}$ Edition
2. Compiler construction D,M.Dhamdhere second edition MACMILLAM.
3. Compiler construction : principles and practices , Kenneth C.Louden ,CENGAGE Learning
4. System software : An introduction to system programming , Leland L. Beck, Pearson

**Assessment:**
**Internal Assessment:**
Assessment consists of two class tests of 20 marks each. The first class test is to be conducted when approx. 40% syllabus is completed and second class test when additional 40% syllabus is completed. Duration of each test shall be one hour.

**End Semester Theory Examination:**
1. Question paper will comprise of 6 questions, each carrying 20 marks.
2. The students need to solve total 4 questions.
3. Question No.1 will be compulsory and based on entire syllabus.
Remaining question (Q.2 to Q.6) will be selected from all the modules

**Online resources**:
- **https://nptel.ac.in/courses/106/108/106108113/**
- **https://nptel.ac.in/courses/106/105/106105190/**
- **https://www.tutorialspoint.com/compiler_design/index.htm**
- **https://www.geeksforgeeks.org/last-minute-notes-compiler-design-gq/**
- **https://pyq.ravindrababuravula.com/subject/?cs=Compiler-Design**
- **https://www.youtube.com/watch?v=oAnkQJT85Ts**
- https://www.youtube.com/watch?v=7Nb-NTGbe-Q&list=PLbRMhDVUMngcseCW7wXDvtTDemCuH80fP&index=2
- https://www.youtube.com/watch?v=xBQEqI91cJM&list=PLbRMhDVUMngcseCW7wXDvtTDemCuH80fP&index=8
- https://www.youtube.com/watch?v=IdCT0KgUZXU&list=PLbRMhDVUMngcseCW7wXDvtTDemCuH80fP&index=17
- https://www.youtube.com/watch?v=QFVorwdIZRg&list=PLbRMhDVUMngcseCW7wXDvtTDemCuH80fP&index=30

**Course objectives:**

1. To understand the role and functioning of various system programs over application program.

2. To understand basic concepts and designing of assembler, Macro processor and role of static and dynamic loaders and linkers.

3. To understand the need to follow the syntax in writing an application program and to learn the how the analysis phase of compiler is designed to understand the programmer's requirements without ambiguity.

4. To synthesize the analysis phase outcomes to produce the object code that is efficient in terms of space and execution time.

**Course Outcomes:**

*Upon completion of this course students will be able to:*

| Identify the relevance of different system programs. | |
|---|---|
| CSC601.1 | Explain the design of two pass assembler . (Understand) |
| CSC601.2 | Explain the design of two pass macroprocessor. (Understand) |
| CSC601.3 | Distinguish between different loaders and linkers schemes. (Understand) |
| CSC601.4 | Explore analysis phase of the compiler. (Understand) |
| CSC601.5 | Describe synthesis phase of compiler (Understand) |

| | PO1 | PO2 | PO3 | PO4 | PO5 | PO6 | PO7 | PO8 | PO9 | PO10 | PO11 | PO12 | PSO1 | PSO2 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CSC601.1 | 1 | 1 | 1 | | | | | | | | | | | |
| CSC601.2 | 1 | 1 | 1 | | | | | | | | | | | |
| CSC601.3 | 1 | | | | | | | | | | | | | |
| CSC601.4 | 1 | 2 | | | 1 | | | | | | | | | |
| CSC601.5 | 1 | 1 | | | | | | | | | | | | |
| Course To PO | | | | | | | | | | | | | | |

**Justification of PO to CO mapping**

| Course Outcome | Competency | Performance Indicator |
|---|---|---|
| CSC601.1 | 1.3 Demonstrate competence in engineering fundamentals | 1.3.1 Apply engineering fundamentals |
| | 2.1 Demonstrate an ability to identify and formulate complex engineering problem | 2.1.3 Identify an algorithm that applies to a given problem |
| | 3.2 Demonstrate an ability to generate a diverse set of alternative design solutions | 3.2.1 Able to explore design alternatives. |
| CSC601.2 | 1.3 Demonstrate competence in engineering fundamentals | 1.3.1 Apply engineering fundamentals |
| | 2.1 Demonstrate an ability to identify and formulate complex engineering problem | 2.1.3 Identify an algorithm that applies to a given problem |
| | 3.2 Demonstrate an ability to generate a diverse set of alternative design solutions | 3.2.1 Able to explore design alternatives. |
| CSC601.3 | 1.3 Demonstrate competence in engineering fundamentals | 1.3.1 Apply engineering fundamentals |
| CSC601.4 | 1.3 Demonstrate competence in engineering fundamentals | 1.3.1 Apply engineering fundamentals |
| CSC601.5 | 1.3 Demonstrate competence in engineering fundamentals | 1.3.1 Apply engineering fundamentals |

**CO Assessment Tools:**

| Course Outcomes | | | | | Indirect Method (20%) | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Unit Tests | | Assignments | | | Quizzes | | | End Sem Exam | Course exit survey |
| | 1 | 2 | 1 | 2 | 3 | 1 | 2 | 3 | | |
| CSC601.1 | | 25% | 25% | | | | -- | | 50% | 100% |
| CSC601.2 | | 25% | | 25% | | | - | | 50% | 100% |
| CSC601.3 | | 15% | | 10% | - | | -- | 25% | 50% | 100% |
| CSC601.4 | 25% | - | - | - | 25% | - | - | | 50% | 100% |
| CSC601.5 | 25% | | | - | | 25% | | | 50% | 100% |

**CO calculation= (0.8 *Direct method + 0.2*Indirect method)**

## <span style="color:red">**Rubrics for the Assignment :**</span>

| Indicator | Very Poor | Poor | Average | Good | Excellent |
|---|---|---|---|---|---|
| **On time Submission (2)** | Assignment not submitted (0) | More than two session late (0.5) | Two sessions late (1) | One session late (1.5) | Early or on time (2) |
| **Organization (2)** | N/A | Very poor readability and not structured (0.5) | Poor readability and somewhat structured (1) | Readable with one or two mistakes and structured (1.5) | Very well written and structured without any mistakes (2) |
| **Level of content (4)** | N/A | Major points are omitted / addressed minimally (1) | All major topics are covered, the information is accurate. (2) | Most major and some minor criteria are included. Information is Accurate (3) | All major and minor criteria are covered and are accurate. (4) |
| **Depth and breadth of discussion (2)** | N/A | None in evidence; superficial at most (0.5) | Minor points/information may be missing and discussion is minimal (1) | Discussion centers on some of the points and covers them adequately (1.5) | Information is presented in depth and is accurate (2) |

# Department of Computer Engineering
## T.E. (Computer) (semester VI)

**Lesson Plan: SPCC**

**Semester VI**                                                 **Year: 2022-23**

**Modes of Content Delivery:**

| I | **Class Room Teaching** | V | **Self Learning Online Resources** | Ix | Industry Visit |
|---|---|---|---|---|---|
| Ii | Tutorial | vi | **Slides** | X | **Group Discussion** |
| Iii | **Remedial Coaching** | vii | Simulations/Demonstrations | xi | **Seminar** |
| Iv | **Lab Experiment** | viii | Expert Lecture | xii | Case Study |

| Lect. No. | Portion to be covered | Planned date | Actual date | Content Delivery Method/Learning Activities |
|---|---|---|---|---|
| 1 | Concept of System Software, Goals of system softwares, system program and system programming, | 11-01-23 | 11-01-23 | Online Teaching, Lab experiment , Self Learning Online Resources |
| 2 | Introduction to various system programs such as Assembler, Macro processor, Loader, Linker, Compiler, Interpreter, Device Drivers, Operating system, Editors, Debuggers. | 12-01-23 | 12-01-23 | Online Teaching, Lab experiment , Self Learning Online Resources |
| 3 | Elements of Assembly Language programming , Pass structure of assembler | 14-01-23 | 14-01-23 | Online Teaching, Lab experiment , Self Learning Online Resources |
| 4 | Assembler Design: Two pass assembler Design and single pass Assembler Design for Hypothetical / X86 family processor, data structures used. | 18-01-23 | 18-01-23 | Class Room Teaching, Lab experiment , Self Learning Online Resources |
| 5 | Assembler Design: Two pass assembler Design and single pass Assembler Design for Hypothetical / X86 family | 21-01-23 | 19-01-23 | Online Teaching, Lab experiment , Self Learning Online Resources |

| | | | | |
|---|---|---|---|---|
| | processor, data structures used | | | |
| 6 | single pass Assembler Design for Hypothetica | 25-01-23 | 21-01-23 | Online Teaching, Lab experiment , Self Learning Online Resources |
| 7 | . Features of Macro facility: Simple, parameterized, conditional, and nested | 28-01-23 | 25-01-23 | Online Teaching, Lab experiment , Self Learning Online Resources |
| 8 | Design of two pass macro processor, data structures used | 01-02-23 | 27-01-23 | Online Teaching, Lab experiment , Self Learning Online Resources |
| 9 | Design of two pass macro processor, data structures used | 02-02-23 | 01-02-23 | Online Teaching, Lab experiment , Self Learning Online Resources |
| 10 | Introduction, functions of loaders, Relocation and Linking concept | 04-02-23 | 02-02-23 | Online Teaching, Lab experiment , Self Learning Online Resources |
| 11 | Direct Linking Loader, | 08-02-23 | 04-02-23 | Online Teaching, Lab experiment , Self Learning Online Resources |
| 12 | Direct Linking Loader, | 09-02-23 | 08-02-23 | Online Teaching, Lab experiment , Self Learning Online Resources |
| 13 | Dynamic linking and loading. | 11-02-23 | 09-02-23 | Online Teaching, Lab experiment , Self Learning Online Resources |
| 14 | Introduction to compilers, Phases of compilers, Lexical Analysis- Role of Finite State Automata in Lexical Analysis | 15-02-23 | 11-02-23 | Online Teaching, Lab experiment , Self Learning Online Resources |
| 15 | Types of Parsers: Top-down parser | 16-02-23 | 16-02-23 | Online Teaching, Lab experiment , Self Learning Online Resources |
| 16 | Design of Predictive parser | 18-02-23 | 17-02-23 | Online Teaching, Lab experiment , Self Learning Online Resources |
| 17 | Design of Predictive parser and LL(1) Grammar | 22-02-23 | 18-02-23 | Online Teaching, Lab experiment , Self Learning Online Resources |
| 18 | Design of recursive descent parser | 23-02-23 | 23-02-23 | Online Teaching, Lab experiment , Self Learning Online Resources |

| 21 | Bottom-up parser- Operator precedence parser | 25-02-23 | 25-02-23 | Online Teaching, Lab Experiment Simulations/Demonstrations |
|---|---|---|---|---|
| 22 | Design of SLR Parser, LR1 | 02-03-23 | 04-03-23 | Online Teaching, Case Study, Lab Experiment |
| 23 | Syntax directed definitions. | 04-03-23 | 09-03-23 | Online Teaching, Case Study, Lab Experiment |
| 24 | Intermediate Code Generation: Types of Intermediate codes: Syntax tree, Postfix notation, Three address codes: Triples and Quadruples. | 08-03-23 | 11-03-23 | Online Teaching, Lab Experiment |
| 25 | Code Optimization: Need and sources of optimization | 09-03-23 | 16-03-23 | Classroom Teaching, PPT |
| 26 | Code optimization techniques: Machine Dependent and Independent | 11-03-23 | 22-03-23 | Classroom Teaching, PPT |
| 27 | Code Generation: Issues in the design of code generator, | 15-03-23 | 29-03-23 | Classroom Teaching, PPT |
| 28 | code generation algorithm. | 16-03-23 | 01-04-23 | Classroom Teaching |
| 29 | Basic block and flow graph. | 22-03-23 | 12-04-23 | Classroom Teaching |
| 30 | University Paper Discussion | 25-03-23 | 13-04-23 | Classroom Teaching, Group Discussion |
| 31 | University Paper Discussion | 29-03-23 | Solution uploaded | Classroom Teaching, Group Discussion |
| 32 | University Paper Discussion | 30-03-23 | Solution uploaded | Classroom Teaching, Group Discussion |
| 33 | University Paper Discussion | 1-04-23 | Solution uploaded | Classroom Teaching, Group Discussion |