ICSE Syllabus



2024

Latest!

Syllabus

aglasem.com

COMPUTER APPLICATIONS (86)

Aims:

1. To empower students by enabling them to build their own applications.
2. To introduce students to some effective tools to enable them to enhance their knowledge, broaden horizons, foster creativity, improve the quality of work and increase efficiency.
3. To develop logical and analytical thinking so that they can easily solve interactive programs.
4. To help students learn fundamental concepts of computing using object oriented approach in one computer language.
5. To provide students with a clear idea of ethical issues involved in the field of computing.

CLASS IX

*There will be **one** written paper of **two hours** duration carrying 100 marks and Internal Assessment of 100 marks.*

The paper will be divided into two sections A and B.

***Section A** (**Compulsory** – 40 marks) will consist of compulsory short answer questions covering the entire syllabus.*

***Section B** (60 marks) will consist of questions which will require detailed answers. There will be a choice of questions in this section.*

THEORY – 100 Marks

1. Introduction to Object Oriented Programming concepts

(i) Principles of Object Oriented Programming, (Difference between Procedure Oriented and Object oriented).
All the four principles of Object Oriented Programming should be defined and explained using real life examples (Data abstraction, Inheritance, Polymorphism, Encapsulation).

(ii) Introduction to JAVA - Types of java programs – Applets and Applications, Java Compilation process, Java Source code, Byte code, Object code, Java Virtual Machine (JVM), Features of JAVA.

Definition of Java applets and Java applications with examples, steps involved in compilation process, definitions of source code, byte code, object code, JVM, features of JAVA - Simple, Robust, secured, object oriented, platform independent, etc.

2. Elementary Concept of Objects and Classes

Modelling entities and their behaviour by objects, a class as a specification for objects and as an object factory, computation as message passing/method calls between objects (many examples should be done to illustrate this). Objects encapsulate state (attributes) and have behaviour (methods). Class as a user defined data type.

A class may be regarded as a blueprint to create objects. It may be viewed as a factory that produces similar objects. A class may also be considered as a new data type created by the user, that has its own functionality.

3. Values and data types

Character set, ASCII code, Unicode, Escape sequences, Tokens, Constants and Variables, Data types, type conversions.

Escape sequences [\n, \t, \\, \", \'], Tokens and its types [keywords, identifiers, literals, punctuators, operators], primitive types and non-primitive types with examples, Introduce the primitive types with size in bits and bytes, Implicit type conversion and Explicit type conversion.

4. Operators in Java

Forms of operators, Types of operators, Counters, Accumulators, Hierarchy of operators, 'new' operator, dot (.) operator.

Forms of operators (Unary, Binary, Ternary), types of operators (Arithmetic, Relational, Logical, Assignment, Increment, Decrement, Short hand operators), Discuss precedence and associativity of operators, prefix and postfix, Creation of dynamic

memory by using new operator, invoking members of class using dot operator, Introduce System.out.println() and System.out.print() – for simple output.

(Bitwise and shift operators are not included).

5. Input in Java

Initialization, Parameter, introduction to packages, Input streams (Scanner Class), types of errors, types of comments

Initialization – Data before execution, Parameters – at the time of execution, input stream – data entry during execution – using methods of Scanner class [nextShort(), nextInt( ), nextLong( ), nextFloat ( ), nextDouble( ), next( ), nextLine( ), next ( ) .charAt(0)]

Discuss different types of errors occurring during execution and compilation of the program (syntax errors, runtime errors and logical errors).Single line comment (//) and multiline comment (/ … */)*

6. Mathematical Library Methods

Introduction to package java.lang [default], methods of Math class.

pow(x,y), sqrt(x), cbrt(x), ceil(x), floor(x), round (x), abs(a), max(a, b), min(a,b), random( ).

Java expressions – using all the operators and methods of Math class.

7. Conditional constructs in Java

Application of if, if else, if else if ladder, switch-case, default, break.

if, if else, if else if, Nested if, switch case, break statement, fall through condition in switch case, Menu driven programs, System.exit(0) - to terminate the program.

8. Iterative constructs in Java

Definition, Types of looping statements, entry controlled loops [for, while], exit controlled loop [do while] , variations in looping statements, and Jump statements.

Syntax of entry and exit controlled loops, break and continue, Simple programs illustrating all three loops, inter conversion from for – while – do while, finite and infinite, delay, multiple counter variables (initializations and updations). Demonstrate break and continue statements with the help of loops.

Loops are fundamental to computation and their need should be shown by examples.

9. Nested for loops

Introduce nested loops through some simple examples. Demonstrate break and continue statements with the help of nested loops.

Programs based on nested loops [rectangular, triangular [right angled triangle only] patterns], series involving single variable.

(Nested while and nested do while are not included.)

10. Computing and Ethics

Ethical Issues in Computing.

Intellectual property rights; protection of individual's right to privacy; data protection on the internet; protection against Spam; software piracy, cybercrime, hacking, protection against malicious intent and malicious code. The stress should be on good etiquette and ethical practices.

INTERNAL ASSESSMENT - 100 Marks

This segment of the syllabus is totally practical oriented. The accent is on acquiring basic programming skills quickly and efficiently.

Programming Assignments (Class IX)

Students are expected to do a minimum of 20 assignments during the whole year to reinforce the concepts studied in the class.

Suggested list of Assignments:

The laboratory assignments will form the bulk of the course. Good assignments should have problems which require design, implementation and testing. They should also embody one or more concepts that have been discussed in the theory class. A significant proportion of the time has to be spent in the laboratory. Computing can only be learnt by doing.

The teacher-in-charge should maintain a record of all the assignments done as a part of practical work throughout the year and give it due credit at the time of cumulative evaluation at the end of the year.

Some sample problems are given below as examples. The problems are of varying levels of difficulty:

(i) Programs using Assignment statements.
Example: Calculation of Area / Volume / Conversion of temperature / Swapping of values etc.

(ii) Programs based on– Input through parameters.
Example: Implementation of standard formula etc.

(iii) Programs based on – Input through Scanner class.
Example: Implementation of standard formula etc.

(iv) Programs based on Mathematical methods.
Example: larger/smaller of two numbers, cube root, square root, absolute value, power, etc.

(v) Programs based on if, if else, if else if ladder, nested if etc.

(a) if programs

- Larger / smaller of two numbers
- To check divisibility of a number, etc.

(b) if - else programs

- Odd or even number
- Eligibility to vote
- Upper case or lower case
- Positive or negative number
- Vowel or Consonant
- Buzz number etc.

(c) if-else-if programs

- Programs based on discount/interest/ bonus/ taxes/ commission.
- Programs based on slab system.
- Programs based on Nested if.

(vi) Programs on switch case.

(a) Day of a week
(b) Name of the month
(c) Names of the seasons
(d) Calculator
(e) Vowel or consonant etc.

(vii) Programs based on Looping Statement

(a) Programs based on for looping statement.

(b) Programs based on printing simple series, summation of simple series, product of simple series.

(c) Prime number, perfect number, composite number, Fibonacci series. Lowest Common Multiple (LCM), Highest Common Factor (HCF) etc.

(d) To find the biggest and smallest number from *n* number of entered numbers.

(e) Program based on while loop like Armstrong number, Spy number, Niven number, Palindrome number, etc.

(viii) Programs based on nested loops [rectangular, triangular(right angled triangle only) patterns], series involving single variable.

(ix) Generate first *n* multiples of numbers from 1 to the limit input by the user.

(x) Menu Driven programs.

Important: This list is indicative only. Teachers and students should use their imagination to create innovative and original assignments.

EVALUATION

Proposed Guidelines for Marking

The teacher should use the criteria below to judge the internal work done. Basically, four criteria are being suggested: class design, coding and documentation, variable description and execution or output. The actual grading will be done by the teacher based on his/her judgment. However, one possible way: divide the outcome for each criterion into one of 4 groups: excellent, good, fair/acceptable, poor/unacceptable, then use numeric values for each grade and add to get the total.

Class design:

Has a suitable class (or classes) been used?
Are all attributes with the right kinds of types present?
Is encapsulation properly done?
Is the interface properly designed?

Coding and Documentation:

Is the coding done properly? (choice of names, no unconditional jumps, proper organization of conditions, proper choice of loops, error handling code layout). Is the documentation complete and readable? (class documentation, variable documentation, method documentation, constraints, known bugs – if any).

Variable and Description

Format for variable description:

Name of the variable	Data Type	Purpose/Description

Evaluation of practical work (Assignments) will be done as follows:

Subject Teacher (Internal Examiner): 100 Marks

Criteria (Total-100 marks)	Class design (20 marks)	Variable description (20 marks)	Coding and Documentation (20 marks)	Execution OR Output (40 marks)
Excellent	20	20	20	40
Good	16	16	16	32
Fair	12	12	12	24
Poor	8	8	8	16

CLASS X

*There will be **one** written paper of **two hours** duration carrying 100 marks and Internal Assessment of 100 marks.*

The paper will be divided into two sections A and B.

***Section A** (**Compulsory** – 40 marks) will consist of compulsory short answer questions covering the entire syllabus.*

***Section B** (60 marks) will consist of questions which will require detailed answers. There will be a choice of questions in this section*

THEORY – 100 Marks

1. Revision of Class IX Syllabus

(i) Introduction to Object Oriented Programming concepts, (ii) Elementary Concept of Objects and Classes, (iii) Values and Data types, (iv) Operators in Java, (v) Input in Java, (vi) Mathematical Library Methods, (vii) Conditional constructs in Java, (viii) Iterative constructs in Java, (ix) Nested for loops.

2. Class as the Basis of all Computation

Objects and Classes

Objects encapsulate state and behaviour – numerous examples; member variables; attributes or features. Variables define state; member methods; Operations/methods/messages/ methods define behaviour.

Classes as abstractions for sets of objects; class as an object factory; primitive data types, composite data types. Variable declarations for both types; difference between the two types. Objects as instances of a class.
Consider real life examples for explaining the concept of class and object.

3. User - defined Methods

Need of methods, syntax of methods, forms of methods, method definition, method calling, method overloading, declaration of methods,

Ways to define a method, ways to invoke the methods – call by value [with programs] and call by reference [only definition with an example], Object creation - invoking the methods with respect to use of multiple methods with different names to implement modular programming, using data members and member methods, Actual parameters and formal parameters, Declaration of methods - static and non-static, method prototype / signature, - Pure and impure methods, - pass by value [with programs] and pass by reference [only definition with an example], Returning values from the methods , use of multiple methods and more than one method with the same name (polymorphism - method overloading).

4. Constructors

Definition of Constructor, characteristics, types of constructors, use of constructors, constructor overloading.
Default constructor, parameterized constructor, constructor overloading., Difference between constructor and method.

5. Library classes

Introduction to wrapper classes, methods of wrapper class and their usage with respect to numeric and character data types. Autoboxing and Unboxing in wrapper classes.

Class as a composite type, distinction between primitive data type and composite data type or class types. Class may be considered as a new data type created by the user, that has its own functionality. The distinction between primitive and composite types should be discussed through examples. Show how classes allow user defined types in programs. All primitive types have corresponding class wrappers. Introduce Autoboxing and Unboxing with their definition and simple examples.

The following methods are to be covered:

int parseInt(String s),

long parseLong(String s),

float parseFloat(String s),

double parseDouble(String s),

boolean isDigit(char ch),

boolean isLetter(char ch),

boolean isLetterOrDigit(char ch),

boolean isLowerCase(char ch),

boolean isUpperCase(char ch),

boolean isWhitespace(char ch),

char toLowerCase (char ch)

char toUpperCase(char ch)

6. Encapsulation

Access specifiers and its scope and visibility.

Access specifiers – private, protected and public. Visibility rules for private, protected and public access specifiers. Scope of variables, class variables, instance variables, argument variables, local variables.

7. Arrays

Definition of an array, types of arrays, declaration, initialization and accepting data of single and double dimensional arrays, accessing the elements of single dimensional and double dimensional arrays.

Arrays and their uses, sorting *techniques - selection sort and bubble sort; Search techniques – linear search and binary search, Array as a composite type, length statement to find the size of the array (sorting and searching techniques using single dimensional array only).*

Declaration, initialization, accepting data in a double dimensional array, sum of the elements in row, column and diagonal elements [right and left], display the elements of two-dimensional array in a matrix format.

8. String handling

String class, methods of String class, implementation of String class methods, String array

The following String class methods are to be covered:

String trim ()

String toLowerCase()

String toUpperCase()

int length( )

char charAt (int n)

int indexOf(char ch)

int lastIndexOf(char ch)

String concat(String str)

boolean equals (String str)

boolean equalsIgnoreCase(String str)

int compareTo(String str)

int compareToIgnoreCase(String str)

String replace (char oldChar,char newChar)

String substring (int beginIndex)

String substring (int beginIndex, int endIndex)

boolean startsWith(String str)

boolean endsWith(String str)

String valueOf(all types)

Programs based on the above methods, extracting and modifying characters of a string, alphabetical order of the strings in an array [Bubble and Selection sort techniques], searching for a string using linear search technique.

INTERNAL ASSESSMENT - 100 Marks

This segment of the syllabus is totally practical oriented. The accent is on acquiring basic programming skills quickly and efficiently.

Programming Assignments (Class X)

The students should complete a minimum of 20 laboratory assignments during the whole year to reinforce the concepts studied in class.

Suggested list of Assignments:

The laboratory assignments will form the bulk of the course. Good assignments should have problems which require design, implementation and testing. They should also embody one or more concepts that have been discussed in the theory class. A significant proportion of the time has to be spent in the laboratory. Computing can only be learnt by doing.

The teacher-in-charge should maintain a record of all the assignments done by the student throughout the year and give it due credit at the time of cumulative evaluation at the end of the year.

Some sample problems are given below as examples. The problems are of varying levels of difficulty:

(i) User defined methods

  (a) Programs depicting the concept of pure, impure, static, non- static methods.

  (b) Programs based on overloaded methods.

(c) Programs involving data members, member methods invoking the methods with respect to the object created.

(ii) Constructors

(a) Programs based on different types of constructors mentioned in the scope of the syllabus.

(b) Programs / outputs based on constructor overloading

(iii) Library classes

(a) Outputs based on all the methods mentioned in the scope of the syllabus.

(b) Programs to check whether a given character is an uppercase/ lowercase / digit etc.

(iv) Encapsulation

Questions based on identifying the different variables like local, instance, arguments, private, public, class variable etc.

(v) Arrays

(a) Programs based on accessing the elements of an array.

(b) Programs based on sort techniques mentioned in the scope of the syllabus.

(c) Programs based on search techniques mentioned in the scope of the syllabus.

(d) Programs on Double dimensional arrays as given in the scope of the syllabus.

(vi) String handling

(a) Outputs based on all the string methods mentioned in the scope of the syllabus.

(b) Programs based on extracting the characters from a given string and manipulating the same.

(c) Palindrome string, pig Latin, alphabetical order of characters, etc.

Important: This list is indicative only. Teachers and students should use their imagination to create innovative and original assignments.

EVALUATION

The teacher-in-charge shall evaluate all the assignments done by the student throughout the year [both written and practical work]. He/she shall ensure that most of the components of the syllabus have been used appropriately in the assignments. Assignments should be with appropriate list of variables and comment statements. The student has to mention the output of the programs.

Proposed Guidelines for Marking

The teacher should use the criteria below to judge the internal work done. Basically, four criteria are being suggested: class design, coding and documentation, variable description and execution or output. The actual grading will be done by the teacher based on his/her judgment. However, one possible way: divide the outcome for each criterion into one of 4 groups: excellent, good, fair/acceptable, poor/unacceptable, then use numeric values for each grade and add to get the total.

Class design:

Has a suitable class (or classes) been used?
Are all attributes with the right kinds of types present?
Is encapsulation properly done?
Is the interface properly designed

Coding and documentation:

Is the coding done properly? (Choice of names, no unconditional jumps, proper organization of conditions, proper choice of loops, error handling, code layout) Is the documentation complete and readable? (class documentation, variable documentation, method documentation, constraints, known bugs - if any).

Variable description:

Format for variable description:

Name of the Variable	Data Type	Purpose/description

Execution or Output:

Does the program run on all sample input correctly?

Evaluation of practical work will be done as follows:

Subject Teacher (Internal Examiner)	50 marks
External Examiner	50 marks

Criteria (Total-50 marks)	Class design (10 marks)	Variable description (10 marks)	Coding and Documentation (10 marks)	Execution OR Output (20 marks)
Excellent	**10**	**10**	**10**	**20**
Good	**8**	**8**	**8**	**16**
Fair	**6**	**6**	**6**	**12**
Poor	**4**	**4**	**4**	**8**

An External Examiner shall be nominated by the Head of the School and may be a teacher from the faculty, but not teaching the subject in the relevant section/class. For example, A teacher of Computer Science of class VIII may be deputed to be the External Examiner for class X.

The total marks obtained out of 100 are to be sent to the Council by the Head of the school.

The Head of the school will be responsible for the online entry of marks on the Council's CAREERS portal by the due date.

EQUIPMENT

There should be enough computer systems to provide for a teaching schedule where at least three-fourth of a time available is used for programming and assignments/practical work. The course shall require at least 4 periods of about 40 minutes duration per week. In one week, out of 4 periods the time should be divided as follows:

- 2 periods – Lecture cum demonstration by the Instructor.
- 2 periods – Assignments/Practical work.

The hardware and software platforms should be such that students can comfortably develop and run programs on those machines.

Since hardware and software evolve and change very rapidly the schools shall need to upgrade them as required. Following are the minimal specifications as of now.

RECOMMENDED FACILITIES:

- A lecture cum demonstration room with a MULTIMEDIA PROJECTOR/ an LCD and Overhead Projector (OHP) attached to the computer.
- A white board with white board markers should be available.
- A fully equipped Computer Laboratory that allows one computer per student.
- The computers should have a minimum of 1 GB RAM and at least a P - IV or Equivalent Processor.
- Good Quality printers.
- A scanner, a web cam/a digital camera (Should be provided if possible).

SOFTWARE FOR CLASSES IX & X

Any suitable Operating System can be used.

For teaching fundamental concepts of computing using object oriented approach, Blue J environment (3.2 or higher version) compatible with JDK (5.0 or higher version) as the base or any other editor or IDE, compatible with JDK (5.0 or higher version) as the base may be used. Ensure that the latest versions of software are used.