

A.S.D.GOVERNMENT DEGREE COLLEGE FOR WOMEN (A)

**Affiliated to Adikavi Nannaya University
Jagannaickpur, Kakinada.**

DEPARTMENT OF COMPUTER SCIENCE



BRIDGE COURSE



2024-2025

A.S.D.GOVERNMENT DEGREE COLLEGE FOR WOMEN (A)

Affiliated to Adikavi Nannaya University
Jagannaickpur, Kakinada.

DEPARTMENT OF COMPUTER SCIENCE

Activity Register 2024-2025

Date	19-08-2024 to 24-08-2024
Conducted through (DRC/JKC/ELF/NCC/NSS/Department etc.,)	Department of Computer Science
Nature of Activity (seminar/workshop/exten Lecture etc)	BRIDGE COURSE I B.Sc. (Computer Science)
Title of the Activity	Fundamentals of Programming
Name of the Department/ Committee	Department of Computer Science
Details of Resourc persons (Name, Designation etc.,)	N.Naga Subrahmanyeswari M.Tech.,(Ph.D). Lecturer in Computer Science K.Surya Lakshmi Guest Lecturer in Computer Science A. Jaya Lakshmi Guest Lecturer in Computer Science
No. of students participated	30
Brief Report on the activity	To get the students acquainted with the Programming Languages and Programmimg Constructs.
Name of the Lecturers who planned & conducted the activity	N.Naga Subrahmanyeswari M.Tech.,(Ph.D). Lecturer in Computer Science K.Surya Lakshmi Guest Lecturer in Computer Science A. Jaya Lakshmi Guest Lecturer in Computer Science
Signature of the Department In-charge/ Convener of the Committee	 IN-CHARGE DEPT OF COMPUTER SCIENCE ASDGOVT DEGREE COLLEGE (W/AUTONOMOUS) KAKINADA
Signature of the Principal	 PRINCIPAL A.S.D.GOV.T.DEGREE COLLEGE (W/A) AUTONOMOUS KAKINADA
Remarks	

PERMISSION LETTER

Kakinada,
Date: 12-08-2024.

To,
Dr. V.Anantha Lakshmi,
Principal,
A.S.D. Govt. Degree College for Women (A),
Kakinada.

From,
N.N.Subrahmanyeswari,
Incharge of Department of Computer Science & Computer Applications,
A.S.D. Govt. Degree College for Women (A),
Kakinada.

Sub: Request to conduct the Bridge Course on “Fundamentals of Programming” for I B.Sc.(CS) students from 19-08-2024 to 24-08-2024-Reg.

Respected Madam,

The Department of Computer Science & Computer Applications wishes to organize the Bridge course on “Fundamentals of Programming” with a duration of 6 days from 19-08-2024 to 24-08-24 for all I B.Sc.(CS) and I B.Com(CA) students. This course aims to make the students get acquainted with the fundamentals of Computer Skills and learn the basics of programming language so that the students will have a developed foundation in the subjects they will encounter, thereby minimizing the challenges of knowledge gaps. Please consider the request to conduct the Bridge course for the students.

Thanking you, Madam.

Yours faithfully,

N.N.S. Eswari
IN-CHARGE
DEPT OF COMPUTER SCIENCE
ASD GOVT DEGREE COLLEGE (W) (AUTONOMOUS)
KAKINADA

CIRCULAR

A.S.D.GOVERNMENT DEGREE COLLEGE FOR WOMEN (A)

Affiliated to Adikavi Nannaya University
Jagannaickpur, Kakinada.

DEPARTMENT OF COMPUTER SCIENCE

CIRCULAR



Date: 13-08-2024

The Department of Computer Science wishes to organize the Bridge Course on “Fundamentals of Programming” from 19-08-2024 to 24-08-2024 for I B.Sc.(Computer Science) students to enhance their Skills on Computers, Basics of Programming language and Programming constructs.

- Timings : 3.00 PM to 4.PM

N.N.S. Eswari
IN-CHARGE
DEPT OF COMPUTER SCIENCE
A.S.D.GOV'T DEGREE COLLEGE (W/AUTONOMOUS)
KAKINADA

Incharge of the Department

V. N. S.
PRINCIPAL
A.S.D.GOV'T.DEGREE COLLEGE (W/A)
AUTONOMOUS
KAKINADA

Principal

A.S.D.GOVERNMENT DEGREE COLLEGE FOR WOMEN (A)

Affiliated to Adikavi Nannaya University

Jagannaickpur, Kakinada.

DEPARTMENT OF COMPUTER SCIENCE

BRIDGE COURSE

on

“Fundamentals of Programming”

The Department of Computer Science conducted Bridge course for I B.Sc (M.P.Cs) students who did have knowledge about Programming Languages and Programming Constructs. With this 6-Day course the students get acquainted with the basic fundamentals of Programming where in the total introduction of the syllabus is covered and there by the student can rise up to a level to apprehend the subject.

OBJECTIVES:

- To be able to learn the basics of various Programming languages used by the Computer.
- To be able to get fundamental knowledge on Programming constructs
- To be able to identify the importance of inculcating the Programming Skills to excel in the field of Computer Science.
- To be able to write basic programs in C.

A.S.D.GOVERNMENT DEGREE COLLEGE FOR WOMEN (A)

Affiliated to Adikavi Nannaya University

Jagannaickpur, Kakinada.

DEPARTMENT OF COMPUTER SCIENCE

ATTENDANCE SHEET

S.No	Name of the Student	19-8-2024	20-8-2024	21-8-2024	22-8-2024	23-8-2024	24-8-2024
1	P. Veera Veni	P	P	A	P	P	A
2	O. Mounika	P	P	P	A	P	P
3	M. Madhavi	A	P	P	P	P	P
4	R. Keerthana	P	A	A	P	P	P
5	R. Durgabhavani	P	A	P	P	P	P
6	P. Harika Satya Devi	A	P	P	A	P	P
7	V. Gouri Sri	P	P	A	P	P	P
8	S. Harika	P	P	A	P	P	P
9	M. Lova Sri	P	A	P	P	P	P
10	T. Durga Bhavani	P	P	P	A	P	P
11	D. Sailaja	P	P	P	A	P	P
12	N. Kerrthi	P	P	A	P	P	P
13	P. Amrutha	P	P	A	P	P	P
14	Sk. Basheeramma	P	A	P	A	P	A
15	S. Rmaya	P	A	P	P	A	P
16	G. Kavya Sri	P	P	A	P	P	P
17	Sk. Ajmanisha	P	P	P	A	P	P
18	V. Teja Sri Surya Kala	P	P	P	P	A	P
19	E. Manju Bhargavi	P	P	A	P	P	P
20	V. Sanjana	P	P	P	A	P	P
21	K. Mounika Kiranmai	P	P	P	A	P	P
22	O. Sindhu Bhairavi	P	P	A	P	P	P
23	K. Lakshmi Sireesha	P	A	P	P	P	P
24	P. Dhana Sri	P	A	P	P	P	P
25	P. Uma Devi	P	P	P	A	P	A
26	D. Divya Jyothi	P	A	P	A	P	P
27	A. Satya Veni	P	P	A	P	P	P
28	D. Lakshmi Ragini	P	P	P	P	P	P
29	U. Mani Kumari	P	A	P	P	P	P
30	P. Ramya	P	P	P	A	P	P

N.N.S. Eswari
IN CHARGE
DEPT. OF COMPUTER SCIENCE
A.S.D.GOV'T DEGREE COLLEGE (WOMEN) KAKINADA
KAKINADA

Signature of the HOD

A.S.D.GOVERNMENT DEGREE COLLEGE FOR WOMEN (A)

Affiliated to Adikavi Nannaya University

Jagannaickpur, Kakinada.

DEPARTMENT OF COMPUTER SCIENCE

BRIDGE COURSE

2024-2025

FUNDAMENTALS OF PROGRAMMING

S.NO	DATE	SYLLABUS
01	19-08-2024 Monday	❖ INTERDUCION TO LANGUAGES
02	20-08-2024 Tuesday	❖ PROGRAMMING LANGUAGES
03	21-08-2024 Wednesday	❖ LANGUAGE TRANSALATORS
04	22-08-2024 Thursday	❖ BASIC PROGRAMMING SYNTACTIC RULES
05	23-08-2024 Friday	❖ TOKENS OF PROGRAMMING LANGUAGES
06	24-08-2024 Saturday	❖ PROGRAMMING CONSTRUCTS

N.N.S. Eswari
INCHARGE
DEPT OF COMPUTER SCIENCE
ASDGOVT DEGREE COLLEGE (W/M/AUT./NONRES)
KAKINADA

Signature of the HOD

A.S.D.GOVERNMENT DEGREE COLLEGE FOR WOMEN (A)

Affiliated to Adikavi Nannaya University

Jagannaickpur, Kakinada.

DEPARTMENT OF COMPUTER SCIENCE



BRIDGE COURSE TIME TABLE

I B.Sc Honours (Computer Science)

2024-2025

DAY	TIMINGS
19-08-2024 Monday	3.00 P.M to 4.00 P.M
20-08-2024 Tuesday	3.00 P.M to 4.00 P.M
21-08-2024 Wednesday	3.00 P.M to 4.00 P.M
22-08-2024 Thursday	3.00 P.M to 4.00 P.M
23-08-2024 Friday	3.00 P.M to 4.00 P.M
24-08-2024 Saturday	3.00 P.M to 4.00 P.M

N.N.S. Eswari
IN CHARGE
DEPT OF COMPUTER SCIENCE
ASD GOVT DEGREE COLLEGE (W) (AUTONOMOUS)
KAKINADA

Signature of the HOD

A.S.D.GOVERNMENT DEGREE COLLEGE FOR WOMEN (A)

Affiliated to Adikavi Nannaya University

Jagannaickpur, Kakinada.

DEPARTMENT OF COMPUTER SCIENCE

BRIDGE COURSE 2024-2025

Fundamentals of Programming

Signature Sheet

S.No	Name of the Student	Class	Student Signature
1	P. Veera Veni	I B.Sc. (CS)	P. Veera Veni
2	O. Mounika	I B.Sc. (CS)	O. Mounika
3	M. Madhavi	I B.Sc. (CS)	M. Madhavi
4	R. Keerthana	I B.Sc. (CS)	A. Sri chandhana
5	R. Durgabhavani	I B.Sc. (CS)	R. Keerthana
6	P. Harika Satya Devi	I B.Sc. (CS)	P. H. Satya Devi
7	V. Gouri Sri	I B.Sc. (CS)	V. Gouri Sri
8	S. Harika	I B.Sc. (CS)	S. Harika
9	M. Lova Sri	I B.Sc. (CS)	M. Lova Sri
10	T. Durga Bhavani	I B.Sc. (CS)	T. Durga Bhavani
11	D. Sailaja	I B.Sc. (CS)	D. Sailaja
12	N. Kerrthi	I B.Sc. (CS)	N. Keerthi
13	P. Amrutha	I B.Sc. (CS)	P. Amrutha
14	Sk. Basheeramma	I B.Sc. (CS)	Sk. Basheeramma
15	S. Rmaya	I B.Sc. (CS)	S. Ramya
16	G. Kavya Sri	I B.Sc. (CS)	G. Kavya Sri
17	Sk. Ajmanisha	I B.Sc. (CS)	Sk. Ajmanisha
18	V. Teja Sri Surya Kala	I B.Sc. (CS)	V. Kala
19	E. Manju Bhargavi	I B.Sc. (CS)	V. T. S. Suryakala
20	V. Sanjana	I B.Sc. (CS)	V. Sanjana
21	K. Mounika Kiranmai	I B.Sc. (CS)	K. Mounika Kiranmai
22	O. Sindhu Bhairavi	I B.Sc. (CS)	O. S. Bhairavi
23	K. Lakshmi Sireesha	I B.Sc. (CS)	K. L. Sireesha
24	P. Dhana Sri	I B.Sc. (CS)	P. Dhana Sri
25	P. Uma Devi	I B.Sc. (CS)	P. Devi
26	D. Divya Jyothi	I B.Sc. (CS)	D. Jyothi
27	A. Satya Veni	I B.Sc. (CS)	A. Satya Veni
28	D. Lakshmi Ragini	I B.Sc. (CS)	D. L. Ragini
29	U. Mani Kumari	I B.Sc. (CS)	U. Mani Kumari
30	P. Ramya	I B.Sc. (CS)	P. Ramya

N. N. S. Eswari
IN CHARGE
DEPT OF COMPUTER SCIENCE
ASD GOVT DEGREE COLLEGE (WOMEN) KAKINADA

Signature of the HOD

A.S.D.GOVERNMENT DEGREE COLLEGE FOR WOMEN (A)

Affiliated to Adikavi Nannaya University

Jagannaickpur, Kakinada.

DEPARTMENT OF COMPUTER SCIENCE

BRIDGE COURSE 2024-2025

Fundamentals of Programming

INTRODUCTION TO LANGUAGES

Programming language

A **programming language** is a **formal language**, which comprises a **set of instructions** that produce various kinds of **output**. Programming languages are used in **computer programming** to implement **algorithms**.

Most programming languages consist of **instructions** for **computers**. There are programmable machines that use a set of **specific instructions**, rather than **general programming languages**. Early ones preceded the **invention of the digital computer**, the first probably being the automatic flute player described in the 9th century by the **brothers Musa** in **Baghdad**, during the **Islamic Golden Age**.^[1] Since the early 1800s, programs have been used to direct the behavior of machines such as **Jacquard looms**, **music boxes** and **player pianos**.^[2] The programs for these machines (such as a player piano's scrolls) did not produce different behavior in response to different inputs or conditions

Types of Programming Language:

There are three types of programming languages

1. Low-level programming language

Low-level language is machine-dependent (0s and 1s) programming language. The processor runs low-level programs directly without the need of a compiler or interpreter, so the programs written in low-level language can be run very fast.

Low-level language is further divided into two parts -

i. Machine Language

Machine language is a type of low-level programming language. It is also called as machine code or object code. Machine language is easier to read because it is normally displayed in binary or hexadecimal form (base 16) form. It does not require a translator to convert the programs because computers directly understand the machine language programs.

The advantage of machine language is that it helps the programmer to execute the programs faster than the high-level programming language.

ii. Assembly Language

Assembly language (ASM) is also a type of low-level programming language that is designed for specific processors. It represents the set of instructions in a symbolic and human-understandable form. It uses an assembler to convert the assembly language to machine language.

The advantage of assembly language is that it requires less memory and less execution time to execute a program.

2. High-level programming language

High-level programming language (HLL) is designed for developing user-friendly software programs and websites. This programming language requires a compiler or interpreter to translate the program into machine language (execute the program).

High-level programming language includes Python, Java, JavaScript, PHP, C#, C++, Objective C, Cobol, Perl, Pascal, LISP, FORTRAN, and Swift programming language.

A high-level language is further divided into three parts -

i. Procedural Oriented programming language

Procedural Oriented Programming (POP) language is derived from structured programming and based upon the procedure call concept.

Procedural Oriented programming language is used by a software programmer to create a program that can be accomplished by using a programming editor like IDE, Adobe Dreamweaver, or Microsoft Visual Studio.

Example: C, FORTRAN, Basic, Pascal, etc.

ii. Object-Oriented Programming language

Object-Oriented Programming (OOP) language is based upon the objects. In this programming language, programs are divided into small parts called objects. It is used to implement real-world entities like inheritance, polymorphism, abstraction, etc in the program to makes the program reusable, efficient, and easy-to-use.

The main advantage of object-oriented programming is that OOP is faster and easier to execute, maintain, modify, as well as debug.

Note: Object-Oriented Programming language follows a bottom-up approach.

Example: C++, Java, Python, C#, etc.

iii. Natural language

Natural language is a part of human languages such as English, Russian, German, and Japanese. It is used by machines to understand, manipulate, and interpret human's language. It is used by developers to perform tasks such as translation, automatic summarization, Named Entity Recognition (NER), relationship extraction, and topic segmentation.

The main advantage of natural language is that it helps users to ask questions in any subject and directly respond within seconds.

3. Middle-level programming language

Middle-level programming language lies between the low-level programming language and high-level programming language. It is also known as the intermediate programming language and pseudo-language.

A middle-level programming language's advantages are that it supports the features of high-level programming, it is a user-friendly language, and closely related to machine language and human language.

Example: C, C++, language

Programming Languages

The Different Programming Languages

- Java and C# Java and C# are two very similar programming languages that are well-optimized and have stricter rules to help prevent programming mistakes. ...
- JavaScript. Since JavaScript runs in all browsers, it can be a good choice of language to learn. ...
- PHP. ...
- Python. ...
- Ruby.

Java and C#

Java and C# are two very similar programming languages that are well-optimized and have stricter rules to help prevent programming mistakes. Code in these languages need to be “compiled” into lower-level code before it runs, and all variables need to be “declared” with their name and type. They also enforce/encourage a methodology known as “object-oriented programming”, requiring all code to belong to an “object”.



People who program in these languages use an IDE to write their software in, which can provide various features to help with programming, such as auto-completion suggestions while they code, and automatic highlighting of certain errors. The rules in these languages will help you detect certain errors before you even run your code, which can be especially helpful when learning programming. However, Java or C# are not made for writing simple scripts, and they are not as popular for quickly creating dynamic websites

JavaScript

Since JavaScript runs in all browsers, it can be a good choice of language to learn. No installation is required, since it can immediately be tried out in the browser. JavaScript can be used for visual effects, but also for doing things without having to update the entire webpage. Modern web apps require JavaScript for many of their features. (For example, try loading Gmail without JavaScript.) JavaScript is also used in many web-related areas, such as creating browser extensions. If you are interested in doing any of these things, Javascript may be a good language to learn. However, Javascript has certain confusing parts, so if you're not planning on using it for the above purposes, you can try a more elegant language such as Python

PHP

PHP is a language built for creating dynamic web pages, and it runs on the server-side. Let's say you just finished [building a website without programming](#) and now you want to be able to customize things further. You want to learn how to program the brains of the website, i.e. the back-end. A large number of websites and scripts are built using PHP, and web hosts often come with a list of one-click-install scripts. If you want to create a plugin for WordPress or work with the same script that runs Wikipedia, then PHP is for you.. This means you should probably learn a different language if you just want to learn programming or you want to create an entirely new web app. However, PHP has improved over time, and if it fits your purposes, go ahead and learn it.

Python

If you just want an easy and elegant language to learn programming, Python is a good choice. Unlike PHP and Javascript, which are made for the web, Python is a general-purpose language that is often used outside of websites. Python aims to be very readable, so even a beginner could figure out what some simple Python code accomplishes. Python has the unusual feature of using indentation to mark different parts of code. This makes the code look less cluttered, but can sometimes cause issues when copying code. Python is a good choice to go with if you don't have a specific goal that fits with one of the other languages.

Ruby

Ruby is similar to Python in many ways. It is a general-purpose language which is focused more on programmer productivity than running-time on a machine. This 'slowness' isn't really an issue for most things a beginner will be building. Ruby has become popular in the last few years due to the website-building framework written in it – Ruby on Rails. Rails developed certain principles (such as “convention over configuration”) that let programmers built websites quickly. If you are interested in creating websites with Rails, then it obviously makes sense to learn some Ruby. While Rails can be used without that much Ruby knowledge, I think a beginner should first learn a simple language before taking on a complex framework.

LANGUAGE TRANSLATORS

Compiling and linking

- Before a program can be executed, three steps are usually necessary: usually necessary:
 - **Preprocessing.** The preprocessor obeys commands that begin with # (known as directives)
 - **Compiling.** A compiler translates then translates the program into machine instructions (object code).
 - **Linking.** A linker combines the object code produced by the compiler with any additional code needed to yield a complete executable program.
- The preprocessor is usually integrated with the compiler. Copyright © 2008 W. W. Norton & Company

Compiler

A **compiler** is a special program that processes statements written in a particular programming language and turns them into machine language or "code" that a computer's processor uses. Typically, a programmer writes language statements in a language such as Pascal or C one line at a time using an editor

Interpreter

In computer science, an *interpreter* is a computer program that directly executes instructions written in a programming or scripting language, without requiring them previously to have been compiled into a machine language program.

DATA TYPES

A data type is a classification of data which tells the compiler or interpreter how the programmer intends to use the data. Most programming languages support various types of data, including integer, real, character or string, and Boolean.^[1]

A major part of understanding how to design and code programs is centered in understanding the types of data that we want to manipulate and how to manipulate that data.

Common data types include:

Data Type	Represents	Examples
integer	whole numbers	-5, 0, 123
floating point (real)	fractional numbers	-87.5, 0.0, 3.14159
string	A sequence of characters	"Hello world!"
Boolean	logical true or false	true, false
nothing	no data	null

The common data types usually exist in most programming languages and act or behave similarly from language to language. Additional complex and/or composite data types may exist and vary from language to language.

TOKENS OF PROGRAMMING LANGUAGES

Tokens

Tokens are the smallest elements of a **program**, which are meaningful to the compiler. The following are the types of **tokens**: Keywords, Identifiers, Constant, Strings, Operators etc.,

There are five categories of **tokens**: 1) identifiers, 2) constants, 3) operators, 4) separators, **and** 5) reserved words. For **example**, the reserved words "new" **and** "function" are **tokens** of the JavaScript language. Operators, such as +, -, *, **and** /, are also **tokens** of nearly all programming languages

Identifiers

Identifiers are names given to various program elements such as variables, functions, and arrays. Identifiers consist of letters and digits, in any order, except that the first character must be a letter. Both uppercase and lowercase letters are permitted and the underscore may also be used, as it is also regarded as a letter. Uppercase and lowercase letters are not equivalent, thus not interchangeable. This is why it is said that C is case sensitive. An identifier can be arbitrarily long. The same identifier may denote different entities in the same program, for example, a variable and an array may be denoted by the same identifier

Constants and Variables

A constant is a value that cannot be changed by the program during normal execution, in other words, the value is constant. When associated with an identifier, a constant is said to be “named,” although the terms “constant” and “named constant” are often used interchangeably. This is contrasted with a variable, which is an identifier with a value that can be changed during normal execution, in other words, the value is variable.

Understanding Constants

A constant is a data item whose value cannot change during the program’s execution. Thus, as its name implies – the value is constant.

A variable is a data item whose value can change during the program’s execution. Thus, as its name implies – the value can vary.

Constants are used in two ways. They are:

1. literal constant
2. defined constant

Literal Constant: A literal constant is a value you type into your program wherever it is needed. Examples include the constants used for initializing a variable and constants used in lines of code:

```
21
12.34
'A'
"Hello world!"
false
null
```

In addition to literal constants, most textbooks refer to symbolic constants or named constants as a constant represented by a name. Many programming languages use ALL CAPS to define named constants.

Language	Example
C++	<code>#define PI 3.14159</code> or <code>const double PI = 3.14159;</code>
C#	<code>const double PI = 3.14159;</code>
Java	<code>const double PI = 3.14159;</code>
JavaScript	<code>Const PI = 3.14159;</code>
Python	<code>PI = 3.14159</code>
Swift	<code>let pi = 3.14159</code>

Technically, Python does not support named constants, meaning that it is possible (but never good practice) to change the value of a constant later. There are workarounds for creating constants in Python, but they are beyond the scope of a first-semester textbook.

Defining Constants and Variables

Named constants must be assigned a value when they are defined. Variables do not have to be assigned initial values. Variables once defined may be assigned a value within the instructions of the program.

Language	Example
C++	<code>double value = 3;</code>
C#	<code>double value = 3;</code>

Java	<code>double value = 3;</code>
JavaScript	<code>var value = 3;let value = 3;</code>
Python	<code>value = 3</code>
Swift	<code>var value:Int = 3</code>

Order of operations

The order of operations (or operator precedence) is a collection of rules that reflect which procedures to perform first in order to evaluate a given mathematical expression

- **Assignment operator**

An assignment statement sets and/or re-sets the value stored in the storage location(s) denoted by a variable name; in other words, it copies a value into the variable

- **Arithmetic Operators**

The basic arithmetic operations are addition, subtraction, multiplication, and division. Arithmetic is performed according to an order of operations

- **Integer Division and Modulus**

In integer division and modulus, the dividend is divided by the divisor into an integer quotient and a remainder. The integer quotient operation is referred to as integer division, and the integer remainder operation is the modulus.¹

- **Unary Operations**

A unary operation is an operation with only one operand. As unary operations have only one operand, they are evaluated before other operations containing them.[1] Common unary operators include Positive (+) and Negative (-).

- **L value and R value**

Some programming languages use the idea of L-values and R-values, deriving from the typical mode of evaluation on the left and right-hand side of an assignment statement. An L-value refers to an object that persists beyond a single expression. An R-value is a temporary value that does not persist beyond the expression that uses it

- **Data Type Conversions**

Changing a data type of a value is referred to as “type conversion”. There are two ways to do this:

1. Implicit – the change is implied
2. Explicit – the change is explicitly done with an operator or function

The value being changed may be:

1. Promotion – going from a smaller domain to a larger domain
2. Demotion – going from a larger domain to a smaller domain

Input-Process-Output Model

The input–process–output (IPO) model is a widely used approach in systems analysis and software engineering for describing the structure of an information processing program or another process. The IPO model is the most basic structure for describing a process.

Separators

A **separator** is a symbol that is used to separate a group of code from one another is called as **separators in java**. In **java**, there are few characters that are used as a **separator**. The most commonly used **separator** is a semicolon. As we have seen it is used to terminate the statement

DECISION MAKING STATEMENTS

Decision making

As in real life, in the virtual world too, decision making involves selecting an option based on certain given conditions. A; programming languages use conditionals or decision-making statements which function based on the following route:

Different programming languages provide different types of decision-making statements. For example, in C programming language, an if...else statement is used when a decision has to be taken where one of two options has to be chosen. Another type of statement which is an alternative to if statements are when the variable has to be tested for equality against certain values whereby each value is called a case and the variable that is switched on is checked for each switch case. A switch is terminated using a break statement.

- **if** statement

Simple if statement

If the *expression* returns true, then the **statement-inside** will be executed, otherwise **statement-inside** is skipped and only the **statement-outside** is executed.

If else statement

If the *expression* is true, the **statement-block1** is executed, else **statement-block1** is skipped and **statement-block2** is executed.

Nested if else statement

if *expression* is false then **statement-block3** will be executed, otherwise the execution continues and enters inside the first **if** to perform the check for the next **if** block, where if *expression 1* is true the **statement-block1** is executed otherwise **statement-block2** is executed.

Else if ladder

The expression is tested from the top(of the ladder) downwards. As soon as a **true** condition is found, the statement associated with it is executed.

- **switch** statement

A **switch** statement allows a variable to be tested for equality against a list of values.

- conditional operator statement (**?:** operator)
- **goto** statement

ITERATION CONTROL STRUCTURES

In iteration control structures, a statement or block is executed until the program reaches a certain state, or operations have been applied to every element of a collection. This is usually expressed with keywords such as `while`, `repeat`, `for`, or `do..until`.

While loop

A "while" loop is a control flow statement that allows code to be executed repeatedly based on a given Boolean condition. The "while" loop can be thought of as a repeating "if" statement.

Do while loop

A do while loop is a control flow statement that executes a block of code at least once, and then repeatedly executes the block, or not, depending on a given boolean condition at the end of the block.^[1]

Some languages may use a different naming convention for this type of loop. For example, the Pascal language has a repeat until loop, which continues to run until the control expression is true (and then terminates) — whereas a “while” loop runs while the control expression is true (and terminates once the expression becomes false)

For loop

A for loop is a control flow statement for specifying iteration, which allows code to be executed repeatedly. A for loop has two parts: a header specifying the iteration, and a body which is executed once per iteration. The header often declares an explicit loop counter or loop variable, which allows the body to know which iteration is being executed. For loops are typically used when the number of iterations is known before entering the loop. For loops can be thought of as shorthands for while loops which increment and test a loop variable.

A.S.D.GOVERNMENT DEGREE COLLEGE FOR WOMEN (A)
Affiliated to Adikavi Nannaya University
Jagannaickpur, Kakinada.

DEPARTMENT OF COMPUTER SCIENCE

BRIDGE COURSE TEST

on

“Fundamentals of Programming”

1. Translator which is used to convert codes of assembly language into machine language is termed as [A]
A.assembler
B.attempter
C.compiler
D.debugger
2. Diagram which shows relationship between classes is termed as [A]
A. Class diagram
B. Sequential diagram
C. Use case diagram
D. Communication diagram
3. Unit which retains processed information until it can be placed on output devices by output unit is [B]
A.input device
B.output device
C.memory unit
D.ALU
4. A computer is a device that can perform computations and make [B]
A.sequential statements
B.Arithmetic statements
C.logical statements
D.algebraic statements
5. Which of the following statements is/are TRUE regarding JAVA ? (a) Constants that cannot be changed are declared using the ‘static’ keyword. (b) A class can only inherit one class but can implement multiple interfaces. [C]
A. Only (a) is TRUE.
B. Only (b) is TRUE.
C. Both (a) and (b) are TRUE.
D. Neither (a) nor (b) are TRUE
6. Which of the following is not an operator in Java? [B]
A. instanceof
B. sizeof
C. New
D. >>>=

7. In Java, after executing the following code what are the values of x, y and z? `int x,y=10; z=12; x=y++ + z++;` [D]
- A. x=22, y=10, z=12
 - B. x=24, y=10, z=12
 - C. x=24, y=11, z=13
 - D. x=22, y=11, z=13
8. In Java, can we make functions inline like C++? [B]
- A. yes
 - B. no
9. What does the following C statement mean?
`scanf("%4s", str);` [A]
- A. Read exactly 4 characters from console.
 - B. Read maximum 4 characters from console.
 - C. Read a string str in multiples of 4
 - D. Nothing
10. Which of the following is true [B]
- A. `gets()` doesn't do any array bound testing and should not be used.
 - B. `fgets()` should be used in place of `gets()` only for files, otherwise `gets()` is fine
 - C. `gets()` cannot read strings with spaces
 - D. None of the above
11. What is the return type of `getchar()`? [A]
- A. Int
 - B. Char
 - C. unsigned char
 - D. Float
12. Which of the following functions from "stdio.h" can be used in place of **printf()**? [D]
- A. `fputs()` with FILE stream as stdout.
 - B. `fprintf()` with FILE stream as stdout.
 - C. `fwrite()` with FILE stream as stdout.
 - D. All of the above three - a, b and c.
 - E. In "stdio.h", there's no other equivalent function of `printf()`
13. Which of the following is not a valid declaration in C? [A]
- 1. `short int x;`
 - 2. `signed short x;`
 - 3. `short x;`
 - 4. `unsigned short x;`
- A. 3 and 4
 - B. 2
 - C. 1
 - D. All are valid

14. In C, sizes of an integer and a pointer must be same. [B]
A. True
B. False
15. Which of the following is not a logical operator? [D]
A. &&
B. !
C. ||
D. |
16. Which of the following can have different meaning in different contexts? [C]
A. &
B. *
C. Both of the above
D. There are no such operators in C
17. #include<stdio.h> [A]
int main()
{
int a = 2,b = 5;
a = a^b;
b = b^a;
printf("%d %d",a,b);
return 0;
}
- A. 52
B. 25
C. 77
D. 7 2
18. Which of the following is true about return type of functions in C? [D]
A. Functions can return any type
B. Functions can return any type except array and functions
C. Functions can return any type except array, functions and union
D. Functions can return any type except array, functions, function pointer and union
19. In C, what is the meaning of following function prototype with empty parameter list [A]
void fun()
{
/* */
}
- A. Function can only be called without any parameter
B. Function can be called with any number of parameters of any types
C. Function can be called with any number of integer parameters.
D. Function can be called with one integer parameter.
20. What is representing the sequence of characters [A]
A. String
B. Integers
C. Floating point
D. Boolean