

**Government Polytechnic College, churu**  
**Department of Computer Science**

**Programme:**Diploma

**Term:** 2018

**Course:** Programming and Problem Solving Through 'C'

**Course CODE:** CS-201

**Class Test:** II

**Year:** IInd

**Max.Marks :15**

**Instructions to candidates:** Attempt all Questions

Q.1 Explain different operators used in C language. (5)

Q.2 Explain various loops in C. (5)

Q.3 What is an array? Explain one dimensional array and two dimensional arrays. (5)

## ANSWERS

### Q.1 ANS

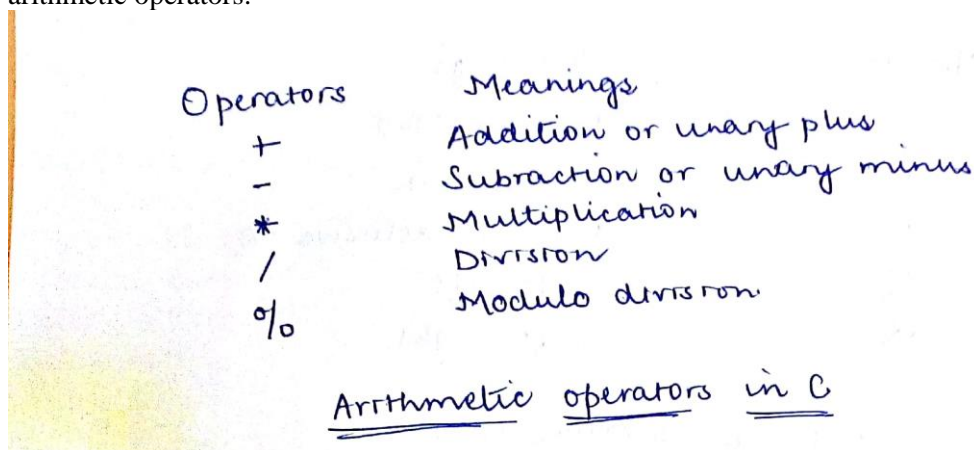
Operators are used in programs to manipulate data and variables.

C operators can be classified into following types:

- Arithmetic operators
- Relational operators
- Logical operators
- Bitwise operators
- Assignment operators
- Conditional operators
- Special operators

### Arithmetic operators

- C supports all the basic arithmetic operators. The following table shows all the basic arithmetic operators.



A handwritten table on a piece of paper with a yellowish tint. The table has two columns: 'Operators' and 'Meanings'. The operators listed are +, -, \*, /, and %. The meanings are: Addition or unary plus, Subtraction or unary minus, Multiplication, Division, and Modulo division. Below the table, the title 'Arithmetic operators in C' is written and underlined.

Operators	Meanings
+	Addition or unary plus
-	Subtraction or unary minus
*	Multiplication
/	Division
%	Modulo division

Arithmetic operators in C

### Relational operators

Operators	Meanings
<	Is less than
<=	Is less than or equal to
>	Is greater than
>=	Is greater than or equal to
==	Is equal to
!=	Is not equal to

Relational operators in C

- **Logical operators**

Operators	Meaning
&&	logical AND
	logical OR
!	logical NOT

Logical operators in C.

- **Bitwise operators**

Bitwise operators perform manipulations of data at **bit level**. These operators also perform **shifting of bits** from right to left. Bitwise operators are not applied to **float** or **double**.

Operators	Meanings
&	Bitwise AND
	Bitwise OR
^	Bitwise exclusive OR
<<	Shift Left
>>	Shift Right

Bitwise operators in C

### **Increment and Decrement Operators**

C programming allows the use of ++ and – operators which are increment and decrement operators respectively. Both the increment and decrement operators are unary operators. The increment operator ++ adds 1 to the operand and the decrement operator – subtracts 1 from the operand. The general syntax of these operators are:

Increment Operator: m++ or ++m;

Decrement Operator: m--or --m;

## Q.2 ANS

**Looping statement** are the statements execute one or more statement repeatedly several number of times.

### Advantage with looping statement

- Reduce length of Code
- Take less memory space.
- Burden on the developer is reducing.
- Time consuming process to execute the program is reduced.

### Types of Loops.

There are three type of Loops available in 'C' programming language.

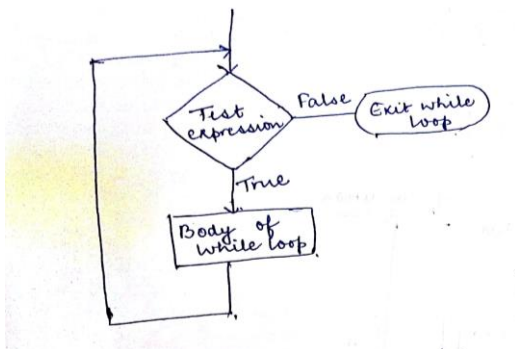
- while loop
- for loop
- do..while

### Difference between conditional and looping statement

Conditional statement executes only once in the program where as looping statements executes repeatedly several number of time.

### While loop

In **while loop** First check the condition if condition is true then control goes inside the loop body other wise goes outside the body. **while loop** will be repeats in clock wise direction.



### Syntax

Assignment;

while(condition)

{

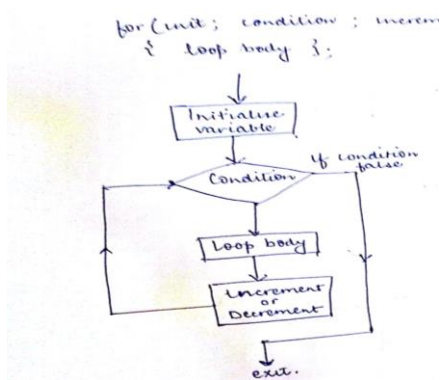
Statements;.....

Increment/decrements (++ or --);

}

### For loop

**for loop** is a statement which allows code to be repeatedly executed. For loop contains 3 parts Initialization, Condition and Increment or Decrements.



### do-while

A **do-while** loop is similar to a while loop, except that a do-while loop is execute at least one time.

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 condition at the end of the block (in while).

### Syntax

do

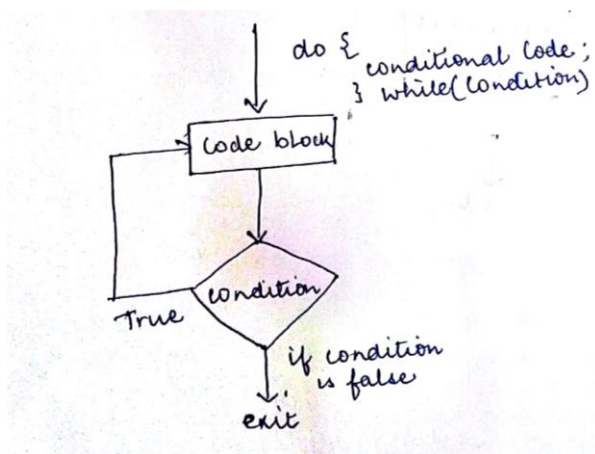
{

Statements;

.....

Increment/decrement (++ or --)

} while();



### Q.3 ANS

- An array is a collection of data items, all of the same type, accessed using a common name.
- A one-dimensional array is like a list; A two dimensional array is like a table; The C language places no limits on the number of dimensions in an array, though specific implementations may.

- Some texts refer to one-dimensional arrays as **vectors**, two-dimensional arrays as **matrices**, and use the general term **arrays** when the number of dimensions is unspecified or unimportant.

### Declaring Arrays

- Array variables are declared identically to variables of their data type, except that the variable name is followed by one pair of square [ ] brackets for each dimension of the array.
- Uninitialized arrays must have the dimensions of their rows, columns, etc. listed within the square brackets.
- Dimensions used when declaring arrays in C must be positive integral constants or constant expressions.

- Examples:

```
int i, j, intArray[ 10 ], number;
float floatArray[ 1000 ];
int tableArray[ 3 ][ 5 ];    /* 3 rows by 5 columns */

const int NROWS = 100;    // ( Old code would use #define NROWS 100 )
const int NCOLS = 200;    // ( Old code would use #define NCOLS 200 )
float matrix[ NROWS ][ NCOLS ];
```

### Initializing Arrays

- Arrays may be initialized when they are declared, just as any other variables.
- Place the initialization data in curly { } braces following the equals sign. Note the use of commas in the examples below.
- An array may be partially initialized, by providing fewer data items than the size of the array. The remaining array elements will be automatically initialized to zero.
- If an array is to be completely initialized, the dimension of the array is not required. The compiler will automatically size the array to fit the initialized data. ( Variation: Multidimensional arrays - see below. )
- Examples:

```
int i = 5, intArray[ 6 ] = { 1, 2, 3, 4, 5, 6 }, k;
float sum = 0.0f, floatArray[ 100 ] = { 1.0f, 5.0f, 20.0f };
double piFractions[ ] = { 3.141592654, 1.570796327, 0.785398163 };
```

### Multidimensional Arrays

- Multi-dimensional arrays are declared by providing more than one set of square [ ] brackets after the variable name in the declaration statement.
- One dimensional arrays do not require the dimension to be given if the array is to be completely initialized. By analogy, multi-dimensional arrays do not require **the first** dimension to be given if the array is to be completely initialized. All dimensions after the first must be given in any case.
- For two dimensional arrays, the first dimension is commonly considered to be the number of rows, and the second dimension the number of columns. We will use this convention when discussing two dimensional arrays.

- Two dimensional arrays are considered by C/C++ to be an array of ( single dimensional arrays ). For example, "int numbers[ 5 ][ 6 ]" would refer to a single dimensional array of 5 elements, wherein each element is a single dimensional array of 6 integers. By extension, "int numbers[ 12 ][ 5 ][ 6 ]" would refer to an array of twelve elements, each of which is a two dimensional array, and so on.
- Another way of looking at this is that C stores two dimensional arrays by rows, with all elements of a row being stored together as a single unit. Knowing this can sometimes lead to more efficient programs.
- Multidimensional arrays may be completely initialized by listing all data elements within a single pair of curly { } braces, as with single dimensional arrays.
- It is better programming practice to enclose each row within a separate subset of curly { } braces, to make the program more readable. This is required if any row other than the last is to be partially initialized. When subsets of braces are used, the last item within braces is not followed by a comma, but the subsets are themselves separated by commas.
- Multidimensional arrays may be partially initialized by not providing complete initialization data. Individual rows of a multidimensional array may be partially initialized, provided that subset braces are used.

**Government Polytechnic College, Churu**  
**Department of Computer Science**

**Programme:** Diploma

**Class Test:** II

**Term:** 2017-18

**Course:** Internet & Web Technologies

**Year:** II

**Course CODE:** CS-209

**Time:** one hour

**Max. Marks :** 15

**Date:** 12-1-2018

**Instructions to candidates:** Attempt All Questions

S. No.	Question	Marks
1.	Explain <SPAN> &<DIV> Tags.	5
2.	Explain PERL language basics.	4
3.	Write Notes(any three):- (a) cookies (b)DHTML (c)java script (d)frames	6



# Answer

## Q.1 Explain <SPAN> & <DIV> Tags.

Ans. The <span> tag is used to group inline-elements in a document, which means that it can be used inside a block element and not create a new line. If you ever want to highlight some text inside a paragraph, wrap that text in a span tag and give it a class. Span tags are often used to incorporate a specific CSS style to differentiate certain parts of content.

Example

```
<style type="text/css">

p { font-size:13pt;font-weight:bold;text-align:justify}

.question { color:brown; font-style:italic}

.big{ font-size:16pt;text-decoration:underline;text-transform:uppercase;color:red}

</style>

<p class="question">how to create <span class="big">style sheet </span>to alter html element?</p>
```

The div element is a block level element, a web page can be divided into segments or division called DIVs. These segments can be positioned anywhere on the page.

Example

```
<DIV ID=box1 style="background-color:red;position:absolute">

<IMG SRC=image/sct-logo.jpg>

</DIV>
```

## Q.2 Explain PERL language basics.

Ans. Perl is a programming language developed by Larry Wall, especially designed for text processing. It stands for Practical Extraction and Report Language. It runs on a variety of platforms, such as Windows, Mac OS, and the various versions of UNIX.

Perl Features

- Perl takes the best features from other languages, such as C, awk, sed, sh, and BASIC, among others.
- Perl's database integration interface DBI supports third-party databases including Oracle, Sybase, Postgres, MySQL and others.
- Perl works with HTML, XML, and other mark-up languages.
- The Perl interpreter can be embedded into other systems.
- Perl supports both procedural and object-oriented programming.

### Q.3 (a) Cookies

An HTTP cookie is a small piece of data sent from a website and stored on the user's computer by the user's web browser while the user is browsing. Cookies were designed to be a reliable mechanism for websites to remember state full information. Cookies information is shared between the client browser and a server using fields in the HTTP headers. If a matching cookie is found among all the stored cookies, the browser sends a cookie field to the server in a request header.

#### (b) DHTML

Dynamic HTML, or DHTML, is an umbrella term for a collection of technologies used together to create interactive and animated websites by using a combination of a static markup language (such as HTML), a client-side scripting language (such as JavaScript), a presentation definition language (such as CSS).

#### (c) JAVA SCRIPT

JavaScript often abbreviated as JS, is a high-level, dynamic, weakly typed, prototype-based, multi-paradigm, and interpreted programming language. Alongside HTML and CSS, JavaScript is one of the three core technologies of World Wide Web content production. It is used to make web pages interactive .

#### (d) FRAMES

The rows attribute of **<frameset>** tag defines horizontal **frames** and cols attribute defines vertical **frames**. Each **frame** is indicated by **<frame>** tag and it defines which **HTML** document shall open into the **frame**. Note – The **<frame>** tag deprecated in HTML5.

Example:

```
<FRAMESET ROWS="33%;33%;33%">
```

```
<FRAMESET COLS="50%;50%">
```

```
</FRAMESET>
```

```
</FRAMESET>
```

**Government Polytechnic College, Churu**  
**Department of Computer Science**

**Programme:** Diploma

**Class Test:** II

**Term:** 2017-18

**Course:** Software Engg.

**Year:** III

**Course CODE:** CS-304

**Time:** one hour

**Max. Marks :** 15

**Date:** 9-1-2018

**Instructions to candidates:** Attempt All Questions

S. No.	Question	Marks
1.	Explain black box testing.	5
2.	Explain software design method.	5
3.	Write Notes(any two):- (a)spiral model (b)software matrix (c)ISO 9000	5

# Answer

## Q.1 Explain black box testing.

Ans. Black-box testing is a method of software testing that examines the functionality of an application without peering into its internal structures or workings. This method of test can be applied virtually to every level of software testing: unit, integration, system and acceptance. Black-box testing is a method of software testing that examines the functionality of an application based on the specifications. It is also known as Specifications based testing. Independent Testing Team usually performs this type of testing during the software testing life cycle.

### BLACK BOX TESTING TECHNIQUES

Following are some techniques that can be used for designing black box tests.

- **Equivalence partitioning:** It is a software test design technique that involves dividing input values into valid and invalid partitions and selecting representative values from each partition as test data.
- **Boundary Value Analysis:** It is a software test design technique that involves determination of boundaries for input values and selecting values that are at the boundaries and just inside/ outside of the boundaries as test data.
- **Cause Effect Graphing:** It is a software test design technique that involves identifying the cases (input conditions) and effects (output conditions), producing a Cause-Effect Graph, and generating test cases accordingly.

## Q.2 Explain software design method.

Ans. Structured methods often support some or all of the following models of a system: A data-flow model, An Entity-relationship model, A structural model, and An object-oriented model. Programs were to be broken into functions and subroutines. There was only a single entry point and a single exit point for any function or routine.

**Function Oriented Design** is decomposed into set of interacting units where each unit has clearly defined function. Conceals the details of an algorithm in a function but system state information is not hidden.

**Object-oriented design** is based on the idea of information hiding. System is viewed as a set of interacting objects, with their own private state.

### Q.3 (a) spiral model

Ans. The **spiral model** is similar to the incremental model, with more emphasis placed on risk analysis. The **spiral model** has four phases: Planning, Risk Analysis, Engineering and Evaluation. A software project repeatedly passes through these phases in iterations called Spirals in this model. The **spiral model**, also known as the spiral lifecycle model, is a systems development method (SDM) used in information technology (IT). This model of development combines the features of the prototyping model and the systems development life cycle (SDLC).

### (b) software matrix

A **software metric** is a standard of measure of a degree to which a software system or process possesses some property. Even if a metric is not a measurement (metrics are functions, while measurements are the numbers obtained by the application of metrics), often the two terms are used as synonyms. Since quantitative measurements are essential in all sciences, there is a continuous effort by computer science practitioners and theoreticians to bring similar approaches to software development. The goal is obtaining objective, reproducible and quantifiable measurements, which may have numerous valuable applications in schedule and budget planning, cost estimation, quality assurance, testing, software debugging, software performance optimization, and optimal personnel task assignments.

### (C) ISO 9000

Ans. **ISO 9000** is a set of international standards on quality management and quality assurance developed to help companies effectively document the quality system elements to be implemented to maintain an efficient quality system. They are not specific to any one industry and can be applied to organizations of any size.

**Government Polytechnic College, Churu**  
**Department of Computer Science**

**Programme:** Diploma

**Class Test:** II

**Term:** 2017-18

**Course:** Data Warehouse & mining

**Year:** III

**Course CODE:** CS-307

**Time:** one hour

**Max. Marks :** 15

**Date:** 11-1-2018

**Instructions to candidates:** Attempt All Questions

S. No.	Question	Marks
1.	Explain data mart types	5
2.	Explain three layer data ware house architecture	5
3.	Write short note (any two):-(a)Decision tree(b)OLAP server(c)Clustering	5

# Answer

**Q.1** Explain data mart types.

**Ans.** Data Mart

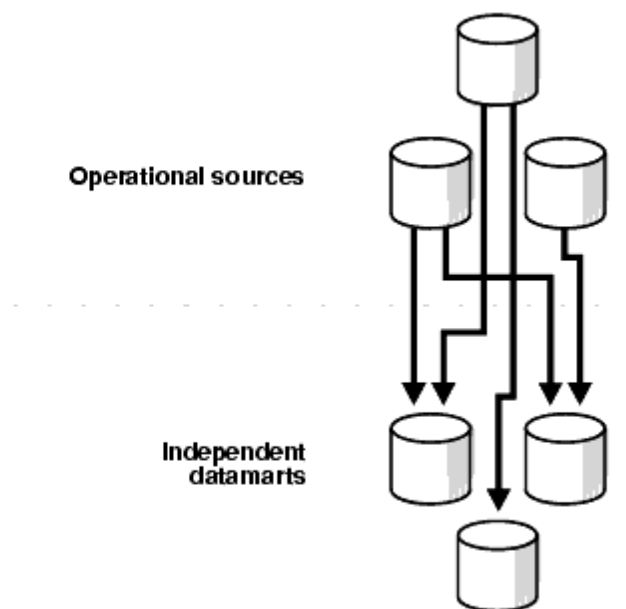
A data mart is a simple form of a data warehouse that is focused on a single subject (or functional area), such as Sales or Finance or Marketing. Data marts are often built and controlled by a single department within an organization. Given their single-subject focus, data marts usually draw data from only a few sources. The sources could be internal operational systems, a central data warehouse, or external data.

There are two types of data marts

1. Independent
2. Dependent

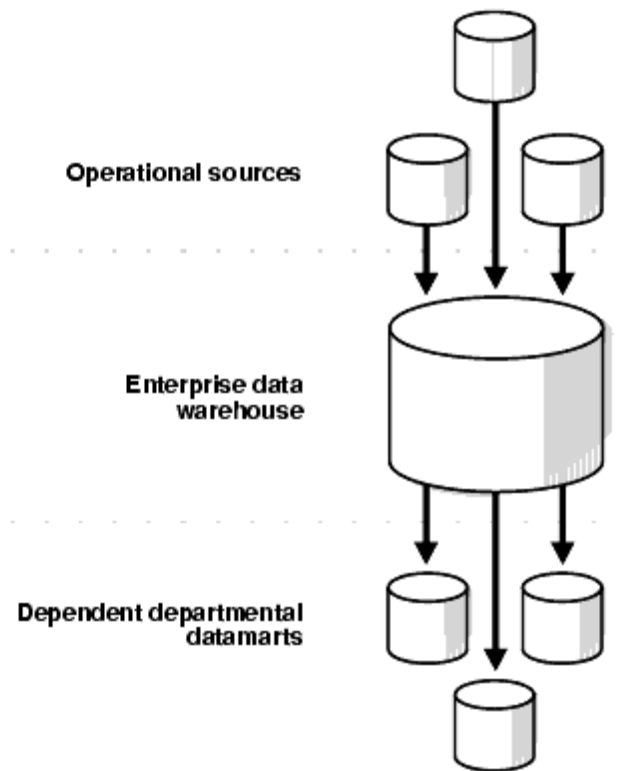
## **1. Independent:-**

An independent data mart is created without the use of a central data warehouse. This could be desirable for smaller groups within an organization. It is not, however, the focus of this Guide. See the Data Mart Suites documentation for further details regarding this architecture. Figure illustrates an independent data mart.



## 2. Dependent:-

A dependent data mart allows you to unite your organization's data in one data warehouse. This gives you the usual advantages of centralization. Illustrates a dependent data mart.



Q.2 Explain three layer data ware house architecture.

Ans. **Three layer Data architecture**

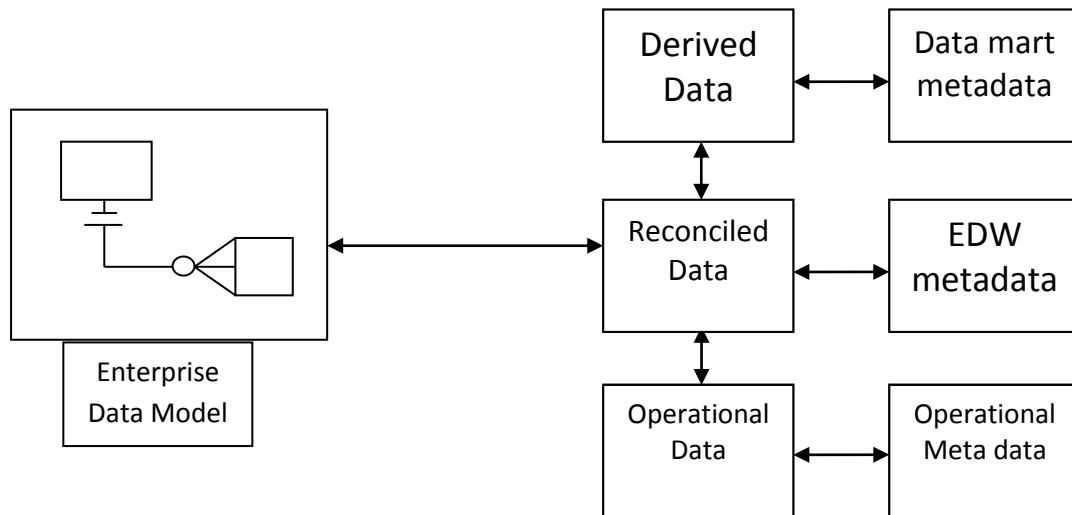
1. Operational data: This is stored in various operational systems of record throughout the organization.
2. Reconciled data: This type of data is stored in and ODS and EDW.
3. Derived data: This type of data is stored in each type of data marts.

This type of architecture is also known as enterprise data model. At every layer of this model we have linked metadata. Metadata are data that describes the characteristics of other data. There are three types of metadata.

- (a) Operational metadata
- (b) EDW (Enterprise Data Warehouse)\
- (c) Data mart metadata

The two critical components of this architecture as shown in fig are enterprise data model and meta data.





### Q.3 (a) Decision tree

Ans. The construction of decision tree classifiers does not require any domain knowledge or parameter setting, and therefore is appropriate for exploratory knowledge discovery. Decision trees can handle high dimensional data. Their representation of acquired knowledge in tree form is intuitive and generally easy to assimilate by humans. The tree classifiers have good accuracy.

### (b) OLAP server

Ans. OLAP stands for online analytical processing system. It is implemented using following two models:

- (a) A relational OLAP (ROLAP) model: It's an extended relational DBMS that maps operations on multidimensional data to standard relational operations.
- (b) A multidimensional OLAP (MOLAP) model: It implements multidimensional data and operations OLAP server

### (C) Clustering

Ans. Clustering, in the context of databases, refers to the ability of several servers or instances to connect to a single database. An instance is the collection of memory and processes that interacts with a database, which is the set of physical files that actually store data.

Clustering offers two major advantages, especially in high-volume database environments:

- Fault tolerance: Because there is more than one server or instance for users to connect to, clustering offers an alternative, in the event of individual server failure.
- Load balancing: The clustering feature is usually set up to allow users to be automatically allocated to the server with the least load.

**Government Polytechnic College, churu**  
**Department of Computer Science**

**Programme:**Diploma

**Term:** 2018

**Course:** Programming and Problem Solving Through 'C'

**Course CODE:** CS-301

**Class Test:** II

**Year:** IInd

**Max.Marks :15**

**Instructions to candidates:** Attempt all Questions

Q.1 What is Linked List? Explain various types of linked lists. (5)

Q.2 Explain Stack and operation performed on Stack. (5)

Q.3 Explain tower of Hanoi problem. (5)

## ANSWERS

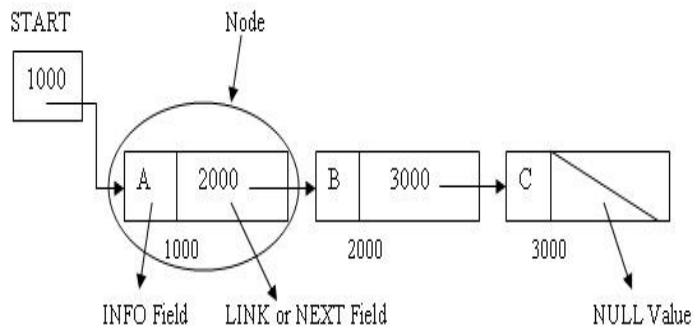
### Q.1 ANS

A linked list can be viewed as a group of items, each of which points to the item in its neighbourhood. An item in a linked list is known as a node. A node contains a data part and one or two pointer part which contains the address of the neighbouring nodes in the list.

Linked list is a data structure that supports dynamic memory allocation and hence it solves the problems of using an array.

#### 1. Linear Linked List or One Way List or Singly Linked List:-

It is linear collection of data elements which are called 'Nodes'. The elements may or may not be stored in consecutive memory locations. So pointers are used maintain linear order. Each node is divided into two parts. The first part contains the information of the element and is called 'INFO Field'. The second part contains the address of the next node and is called 'LINK Field' or 'NEXT Pointer Field'. The START contains the starting address of the linked list i.e. it contains the address of the first node of the linked list. The LINK Field of last node contains NULL Value which indicates that it is the end of linked list. It is shown below:



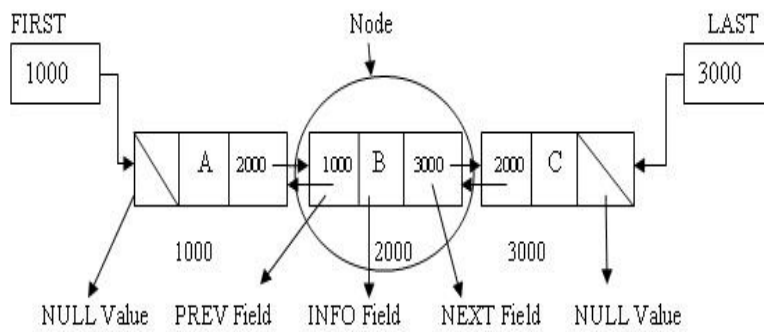
#### 2. Doubly Linked List or Two-Way Linked List or Two-Way Chain:-

In it each node is divided into three parts:

1. The first part is PREV part. It is previous pointer field. It contains the address of the node which is before the current node.
2. The second part is the INFO part. It contains the information of the element.
3. The third part is NEXT part. It is next pointer field. It contains the address of the node which is after the current node.

There are two pointers FIRST and LAST. FIRST points to the first node in the list. LAST points to the last node in the list. The PREV field of first node and the NEXT field of last node contain NULL value. This shows the end of list on both sides. This list can be traversed in both directions

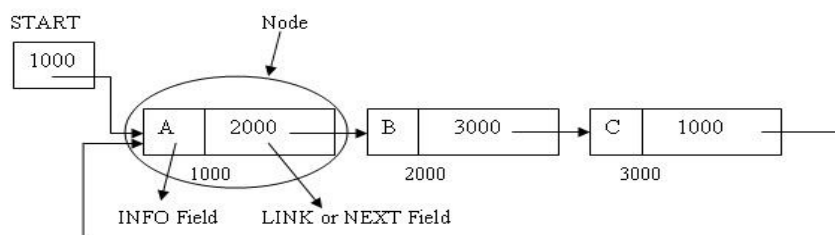
that is forward and backward.



### 3. Circular Linked List:-

In it the last node does not contain NULL pointer. Instead the last node contains a pointer that has the address of first node and thus points back to the first node.

It is shown below:



## Q.2 ANS

Stack is an abstract data type with a bounded (predefined) capacity. It is a simple data structure that allows adding and removing elements in a particular order. Every time an element is added, it goes on the top of the stack, the only element that can be removed is the element that was at the top of the stack, just like a pile of objects.

### Basic features of Stack

- Stack is an ordered list of similar data type.
- Stack is a **LIFO** structure. (Last in First out).
- **push()** function is used to insert new elements into the Stack and **pop()** function is used to delete an element from the stack. Both insertion and deletion are allowed at only one end of Stack called **Top**.
- Stack is said to be in **Overflow** state when it is completely full and is said to be in **Underflow** state if it is completely empty.

### Applications of Stack

The simplest application of a stack is to reverse a word. You push a given word to stack - letter by letter - and then pop letters from the stack.

There are other uses also like : **Parsing, Expression Conversion**(Infix to Postfix, Postfix to Prefix etc) and many more.

### Implementation of Stack

Stack can be easily implemented using an Array or a Linked List. Arrays are quick, but are limited in size and Linked List requires overhead to allocate, link, unlink, and deallocate, but is not limited in size. Here we will implement Stack using array.

#### Algorithm for PUSH operation

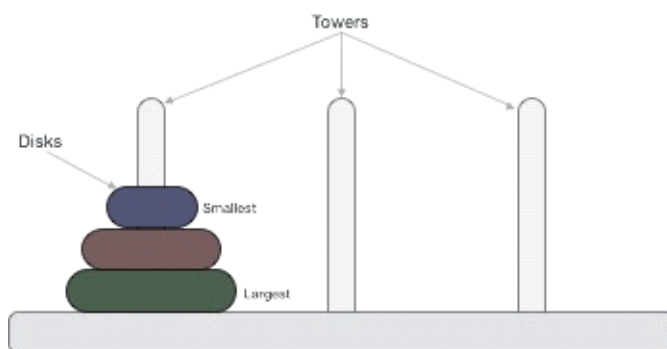
- Check if the stack is full or not.
- If the stack is full, then print error of overflow and exit the program.
- If the stack is not full, then increment the top and add the element .

#### Algorithm for POP operation

- Check if the stack is empty or not.
- If the stack is empty, then print error of underflow and exit the program.
- If the stack is not empty, then print the element at the top and decrement the top.

### Q.3 ANS

Tower of Hanoi, is a mathematical puzzle which consists of three towers (pegs) and more than one rings.



These rings are of different sizes and stacked upon in an ascending order, i.e. the smaller one sits over the larger one. There are other variations of the puzzle where the number of disks increase, but the tower count remains the same.

### Rules

The mission is to move all the disks to some another tower without violating the sequence of arrangement. A few rules to be followed for Tower of Hanoi are –

- Only one disk can be moved among the towers at any given time.
- Only the "top" disk can be removed.
- No large disk can sit over a small disk.

**Government Polytechnic College, churu**

**Department of Computer Science**

**Programme:**Diploma

2018

**Course:** Programming and Problem Solving Through 'C'

**Course CODE:** CS-309

**Instructions to candidates:** Attempt all Questions

**Class Test: II Term:**

**Year:** IInd

**Max.Marks :15**

Q.1 What is the difference between Application and Applet? (3)

Q.2 Explain applet life cycle (4)

Q.3 Explain commonly used methods of Graphics class (4)

Q.4 Explain layout manager. (4)

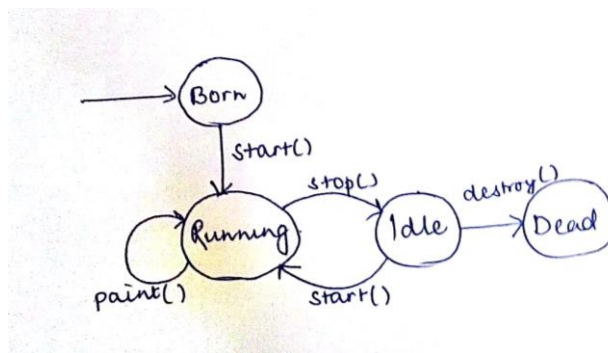
## ANSWERS

### Q.1 ANS

1. Applets don't use the main() method, but when they are load, automatically call certain methods(init, start, paint, stop, destroy).
2. They are embedded inside a web page and executed in browsers.
3. They cannot read from or write to the files on local computer.
4. They cannot communicate with other servers on the network.
5. They cannot run any programs from the local computer.
6. They are restricted from using libraries from other languages.

### Q.2 ANS

Java applet inherits features from the class Applet. Thus, whenever an applet is created, it undergoes a series of changes from initialization to destruction. Various stages of an applet life cycle are depicted in the figure below:



**Applet life cycle**

**init():** Which will be executed whenever an applet program start loading, it contains the logic to initiate the applet properties.

**start():** It will be executed whenever the applet program starts running.

**stop():** Which will be executed whenever the applet window or browser is minimized.

**destroy():** It will be executed whenever the applet window or browser is going to be closed (at the time of destroying the applet program permanently).

Apart from the above stages, Java applet also possess paint( ) method. This method helps in drawing, writing and creating colored backgrounds of the applet. It takes an argument of the graphics class. To use The graphics, it imports the package `java.awt.Graphics`

### Q.3 ANS

Commonly used methods of Graphics class:

- **public abstract void drawString(String str, int x, int y):** is used to draw the specified string.
- **public void drawRect(int x, int y, int width, int height):** draws a rectangle with the specified width and height.
- **public abstract void fillRect(int x, int y, int width, int height):** is used to fill rectangle with the default color and specified width and height.
- **public abstract void drawOval(int x, int y, int width, int height):** is used to draw oval with the specified width and height.
- **public abstract void fillOval(int x, int y, int width, int height):** is used to fill oval with the default color and specified width and height.
- **public abstract void drawLine(int x1, int y1, int x2, int y2):** is used to draw line between the points(x1, y1) and (x2, y2).
- **public abstract boolean drawImage(Image img, int x, int y, ImageObserver observer):** is used draw the specified image.
- **public abstract void drawArc(int x, int y, int width, int height, int startAngle, int arcAngle):** is used draw a circular or elliptical arc.
- **public abstract void fillArc(int x, int y, int width, int height, int startAngle, int arcAngle):** is used to fill a circular or elliptical arc.
- **public abstract void setColor(Color c):** is used to set the graphics current color to the specified color.
- **public abstract void setFont(Font font):** is used to set the graphics current font to the specified font.

### Q.4 ANS

A layout manager automatically arranges the controls within a window by using some type of algorithm. In GUI environments, such as Windows, you can layout your controls by hand. It is possible to lay out Java controls by hand, too, but we do not do so for two main reasons. First, it is very tedious to manually lay out a large number of components. Second, sometimes the width and height information is not yet available when you need to arrange some control, because the native toolkit components have not been realized. This is a chicken-and-egg situation; it is confusing to figure out when it is okay to use the size of a given component to position it relative to another.

Each Container object has a layout manager associated with it. A layout manager is an instance of any class that implements the `LayoutManager` interface. The layout manager is set by the `setLayout()` method. If no call to `setLayout()` is made, then the default layout manager is used. Whenever a container is resized or sized for the first time, the layout manager is used to position each of the components within it.

The `setLayout()` method has the following general form:



```
void setLayout(LayoutManager layoutObj)
```

Here, layoutObj is a reference to the desired layout manager. If you wish to disable the layout manager and position components manually, pass null for layoutObj. If you do this, you will need to determine the shape and position of each component manually, using the setBounds( ) method defined by Component.

Each layout manager keeps track of a list of components that are stored by their names. The layout manager is notified each time you add a component to a container. Normally, you will want to use a layout manager. Whenever the container needs to be resized, the layout manager is consulted via its minimumLayoutSize( ) and preferredLayoutSize( ) methods. Each component that is being managed by a layout manager contains the getPreferredSize( ) and getMinimumSize( ) methods. These return the preferred and minimum size required to display each component.. You may override these methods for controls that you subclass. Default values are provided otherwise.

Java has several predefined LayoutManager classes.

- FlowLayout
- BorderLayout
- Insets
- GridLayout
- CardLayout

### **FlowLayout**

FlowLayout is the default layout manager. FlowLayout implements a simple layout style, which is similar to how words flow in a text editor. Components are laid out from the upper-left corner, left to right and top to bottom. When no more components fit on a line, the next one appears on the next line. A small space is left between each component, above and below, as well as left and right.

The constructors for FlowLayout are shown below:

- FlowLayout( )
- FlowLayout(int how)
- FlowLayout(int how, int horz, int vert)

The first form creates the default layout, which centers components and leaves five pixels of space between each component.

The second form lets you specify how each line is aligned. Valid values for how are as follows:

- FlowLayout.LEFT
- FlowLayout.CENTER
- FlowLayout.RIGHT

These values specify left, center, and right alignment, respectively. The third form allows specifying the horizontal and vertical space left between components in horz and vert, respectively.

**Government Polytechnic College, churu**  
**Department of Computer Science**

**Programme:**Diploma

**Term:** 2018

**Course:** Programming and Problem Solving Through 'C'

**Course CODE:** EI-210

**:15**

**Class Test: II**

**Year: IInd**

**Max.Marks**

**Instructions to candidates:** Attempt all Questions

- Q.1 Explain different operators used in C language. (4)
- Q.2 Explain SWITCH CASE IN C. (3)
- Q.3 Differentiate between while and do-while loop (3)
- Q.4 What is an array? Explain one dimensional array and two dimensional array. (5)

## ANSWERS

### Q.1 ANS

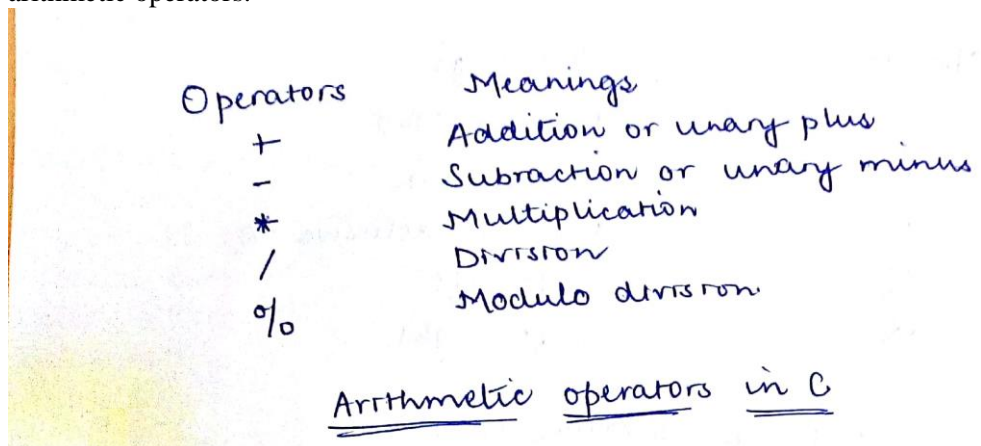
Operators are used in programs to manipulate data and variables.

C operators can be classified into following types:

- Arithmetic operators
- Relational operators
- Logical operators
- Bitwise operators
- Assignment operators
- Conditional operators
- Special operators

### Arithmetic operators

- C supports all the basic arithmetic operators. The following table shows all the basic arithmetic operators.



A handwritten table with two columns: 'Operators' and 'Meanings'. The operators listed are +, -, \*, /, and %. The meanings are: Addition or unary plus, Subtraction or unary minus, Multiplication, Division, and Modulo division. Below the table, the title 'Arithmetic operators in C' is underlined.

Operators	Meanings
+	Addition or unary plus
-	Subtraction or unary minus
*	Multiplication
/	Division
%	Modulo division

Arithmetic operators in C

### Relational operators

Operators	Meanings
<	Is less than
<=	Is less than or equal to
>	Is greater than
>=	Is greater than or equal to
==	Is equal to
!=	Is not equal to

Relational operators in C

- **Logical operators**

Operators	Meaning
&&	logical AND
	logical OR
!	logical NOT

Logical operators in C.

- **Bitwise operators**

Bitwise operators perform manipulations of data at **bit level**. These operators also perform **shifting of bits** from right to left. Bitwise operators are not applied to **float** or **double**.

Operators	Meanings
&	Bitwise AND
	Bitwise OR
^	Bitwise exclusive OR
<<	Shift Left
>>	Shift Right

Bitwise operators in C

### **Increment and Decrement Operators**

C programming allows the use of ++ and – operators which are increment and decrement operators respectively. Both the increment and decrement operators are unary operators. The increment operator ++ adds 1 to the operand and the decrement operator – subtracts 1 from the operand. The general syntax of these operators are:

Increment Operator: m++ or ++m;

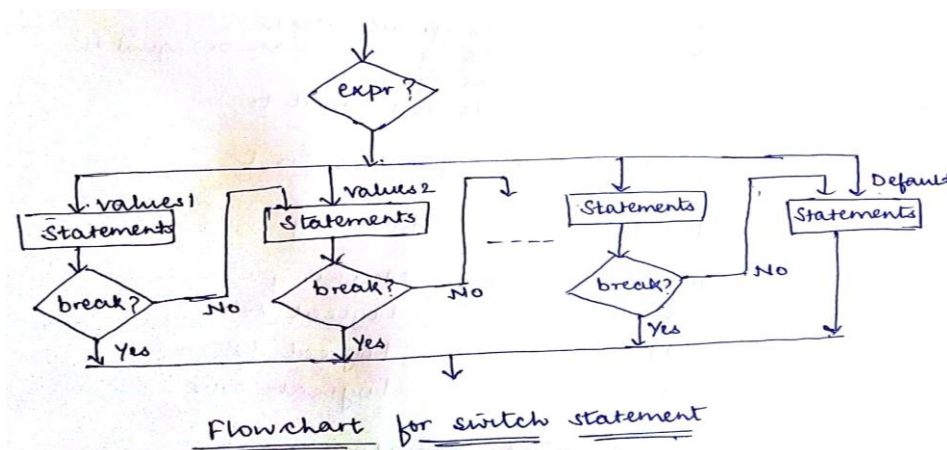
Decrement Operator: m--or --m;

## Q.2 ANS

The switch **statement** allows us to test an expression and depending on its value, execute a statement (or group of statements) amongst several alternatives. It is the most involved statement provided in C language. It uses four keywords, namely, switch, case, default and break. The last two keywords are optional and can be omitted. The general form of this statement is given below and its flowchart is shown in.

```
switch ( expr )
{
    case value1 :
        statement;
        statement;
        .....
        break;
    case value2:
        statement;
        statement;
        break;
    .....
    default:
        statement;
        statement;
        break;
}
```

The test expression expr is written within parentheses after the switch keyword. The value of this expression is used to select one of the alternative groups of statements (cases), written using the case and default keywords, for execution. Note that the test expression must be an integral expression. A beginner often makes the mistake of using either a floating or a string expression as a test expression in a switch statement.



## Q.2 ANS

While loop	Do... while loop
Syntax: while(condition), {	Syntax: do, {
}	} while(condition);
Semicolon (;) is not used	Semicolon (;) is used
Condition is checked first	Condition is checked later.
Since condition is checked first, statements may or may not get executed.	Since condition is checked later, the body statements will execute at least once.
The main feature of the while loop is, it's an entry controlled loop.	The main feature of the do while loops is it is an exit controlled loop

## Q.3 ANS

- An array is a collection of data items, all of the same type, accessed using a common name.
- A one-dimensional array is like a list; A two dimensional array is like a table; The C language places no limits on the number of dimensions in an array, though specific implementations may.
- Some texts refer to one-dimensional arrays as **vectors**, two-dimensional arrays as **matrices**, and use the general term **arrays** when the number of dimensions is unspecified or unimportant.

### Declaring Arrays

- Array variables are declared identically to variables of their data type, except that the variable name is followed by one pair of square [ ] brackets for each dimension of the array.
- Uninitialized arrays must have the dimensions of their rows, columns, etc. listed within the square brackets.
- Dimensions used when declaring arrays in C must be positive integral constants or constant expressions.
- Examples:

```
int i, j, intArray[ 10 ], number;  
float floatArray[ 1000 ];  
int tableArray[ 3 ][ 5 ];    /* 3 rows by 5 columns */
```

```
const int NROWS = 100;    // ( Old code would use #define NROWS 100 )  
const int NCOLS = 200;    // ( Old code would use #define NCOLS 200 )  
float matrix[ NROWS ][ NCOLS ];
```

### Initializing Arrays

- Arrays may be initialized when they are declared, just as any other variables.
- Place the initialization data in curly { } braces following the equals sign. Note the use of commas in the examples below.

- An array may be partially initialized, by providing fewer data items than the size of the array. The remaining array elements will be automatically initialized to zero.
- If an array is to be completely initialized, the dimension of the array is not required. The compiler will automatically size the array to fit the initialized data. ( Variation: Multidimensional arrays - see below. )

- Examples:

```
int i = 5, intArray[ 6 ] = { 1, 2, 3, 4, 5, 6 }, k;
float sum = 0.0f, floatArray[ 100 ] = { 1.0f, 5.0f, 20.0f };
double piFractions[ ] = { 3.141592654, 1.570796327, 0.785398163 };
```

## Multidimensional Arrays

- Multi-dimensional arrays are declared by providing more than one set of square [ ] brackets after the variable name in the declaration statement.
- One dimensional arrays do not require the dimension to be given if the array is to be completely initialized. By analogy, multi-dimensional arrays do not require **the first** dimension to be given if the array is to be completely initialized. All dimensions after the first must be given in any case.
- For two dimensional arrays, the first dimension is commonly considered to be the number of rows, and the second dimension the number of columns. We will use this convention when discussing two dimensional arrays.
- Two dimensional arrays are considered by C/C++ to be an array of ( single dimensional arrays ). For example, "int numbers[ 5 ][ 6 ]" would refer to a single dimensional array of 5 elements, wherein each element is a single dimensional array of 6 integers. By extension, "int numbers[ 12 ][ 5 ][ 6 ]" would refer to an array of twelve elements, each of which is a two dimensional array, and so on.
- Another way of looking at this is that C stores two dimensional arrays by rows, with all elements of a row being stored together as a single unit. Knowing this can sometimes lead to more efficient programs.
- Multidimensional arrays may be completely initialized by listing all data elements within a single pair of curly { } braces, as with single dimensional arrays.
- It is better programming practice to enclose each row within a separate subset of curly { } braces, to make the program more readable. This is required if any row other than the last is to be partially initialized. When subsets of braces are used, the last item within braces is not followed by a comma, but the subsets are themselves separated by commas.
- Multidimensional arrays may be partially initialized by not providing complete initialization data. Individual rows of a multidimensional array may be partially initialized, provided that subset braces are used.

