# 

# C++ Programming

## TYPES

### Data types

#### Integer types

Limits for integer types:

Internal implementation:

Why 2’s complement is used widely?

#### Float types:

Internal implemenatation:

FAQ

How to find the number is negative when it is represented using 2’s complement?

#### Character types:

#### Function type

Stdint header file:

Need for the stdint:

Recursive (or) Inductive Data type

TYPE conversion in c++

integer to unsigned integer

## INITIALIZATION

#### Types of initialization

Zero Initialization

User defined type

Syntax

Standard type

Default initialization (In terms of taking the initialization values)

Value initialization

Standard types

1. Use the values in curly braces

#### Initializer list

#### 

#### List Initialization

Using list initialization in constructor

Initializer\_list class

Map list initialization

### storage class specifier IN C++

Auto:

STATIC:

Static keyword for data members for class

REGISTER:

Extern

Thread\_local

GNU Compiler:

Mutable:

ACCESS SPECIFIER IN C++

#### Storage class specifier puzzles

1. Whether data members of the class can be initialized at declaration time?

OUTPUT:

EXPLANATION:

1. Whether const static data members of the class has to be redefined outside the class declaration as other static data members of the class?

OUTPUT:

EXPLANATION:

1. Whether static function can be called by using the objects of the class?

OUTPUT:

EXPLANATION:

1. How to access the global variable when local variable has same name as global variable?

OUTPUT:

EXPLANATION:

### OPERATOR IN C++

SCOPE RESOLUTION OPERATOR:

TYPEDEF IN C++

Difference between Data type, Data Structure and Abstract data type

### POINTERS:

TERMINOLOGY

STRICT ALIASING RULE:

Printing pointers

Why printing pointers by directly assigning some integer value to pointer returning different value ?

#### Pointer puzzles

1. What happens when we access the single pointer with double array braces?

OUTPUT:

EXPLANATION:

### REFERENCE

Relation with array

Reference to array

Syntax

Array of references

It is not possible in c++

#### REFERENCE PUZZLES

1. Whether same reference variable can be assigned with other variable after declaration?

OUTPUT:

EXPLANATION:

### VALUE CATEGORY

Rvalue

### IMPLICIT TYPE CONVERSION

Allowed types overview

### EXPLICIT TYPE CONVERSION

#### CASTING

Metaphor

What is need for casting?

Why C++ Style casting?

Why static and dynamic cast is named so?

Implicit conversion and casting

Types of casting operators

Type cast operator (or) user defined conversion operator

#### TYPES OF CASTING

Metaphor

Static cast

Metaphor

Dynamic cast

Polymorphic class:

What happens when invalid class is casted?

Metaphor

Reinterpret\_cast

### Reference variable in c++

Datatype conflict:

Difference between pointer and references:

REFERENCE VARIABLE IN C:

### STRING

String is implemented as class in c++.

Size\_type

npos

Input output class

Class Structure

Nested class

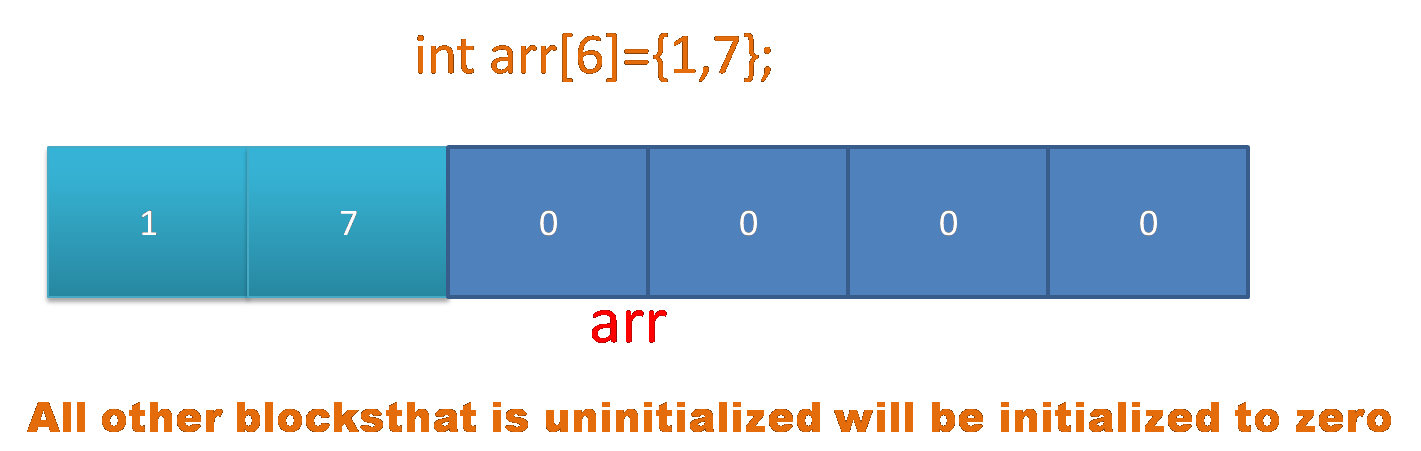
Why nested class?

String literal

|  |  |
| --- | --- |
|  |  |
|  |  |

### Array

#### Initialization of array



#### Flavours of array size

VARIABLE LENGTH ARRAY

Tail padded array or Zero size array

Flexible array member

#### ARRAY RELATION WITH POINTERS:

When array will be decompose in to pointers?

#### Internal representation of array:

Array braces

#### Array puzzles

1. What is the size of the two dimensional array when some integer is added to it?

OUTPUT:

EXPLANTION:

1. Can we do an parenthesis based initialization for dynamic array?

OUTPUT:

EXPLANATION:

1. what is size of the two dimensional array when dereference operator was used?

OUTPUT:

EXPLANATION:

1. What happens when we use increment operator on a array parameter in function?

OUTPUT:

EXPLANATION:

1. Whether size can be given as null for array?

OUTPUT:

EXPLANATION:

ENUM

## CONDITIONAL AND LOOPING STATEMENTS

### IF STATEMENT

#### FAQ

1. What happen when we assign variable inside the if statement?

### Goto

#### GOTO PUZZLES

1. Whether label which is declared in other functions can be accessed in the current function?

OUTPUT:

EXPLANATION:

1. Whether labels which are declared within the inner block scope of function can be accessed by the same function?

OUTPUT:

EXPLANATION:

## OPERATORS

### General Operators

#### COMMA OPERATOR

OPERATOR Vs SEPARATOR

Associativity & Precedence:

#### BITWISE operator

Frequently asked questions:

1. How to compare whether value is zero or 1 by using bit wise operator?

#### Placement New operator

#### MISCELLANEOUS

Whether semicolon is operator in C++?

#### General OperatorS Puzzles

1. Priority of comma operator between equal operator and brackets?

OUTPUT:

EXPLANATION:

1. Can we place the array variable inside the array subscript operator?

OUTPUT

EXPLANATION:

1. Can we use the array subscript operator for char literal without assigning in variable?

OUTPUT:

EXPLANATION:

1. What is operator precedence among the pre-increment, post-increment and indirection operator?

OUTPUT:

EXPLANATION:

1. What is operator precedence among the pre-increment, post-increment and indirection operator during operator overloading?

OUTPUT:

EXPLANATION:

1. Consider post increment operator is overloaded, then when it is called when other overloaded operators are used in the expression?

OUTPUT:

EXPLANATION:

## FUNCTIONS

### Terminology

Helper function:

GLOBAL FUNCTIONS:

Variable Arguments:

Mutually recursive function

#### FUNCTION PUZZLES

1. What happens when variable is initialized with function which returns void?

OUTPUT:

EXPLANATION:

### FUNCTION POINTERS

RETURNING FUNCTION POINTER:

#### FUNCTION POINTER PUZZLES

1. What happen when we dereference the function pointer?

OUTPUT:

EXPLANATION:

1. How to return the function?

OUTPUT:

EXPLANATION:

### Exception handling

No throw statement

#### STANDARD EXCEPTIONS

IMPLEMENATION DETAILS

MEMORY RELATED EXCEPTION

bad\_alloc

When memory allocation is failed, then bad\_alloc exception will be thrown.

### VIRTUAL FUNCTION IN C++

Note

When we need to use the virtual function?

How virtual function is called at run time?

Statically Typed and DynamicallyTyped languages

Early/compile-time binding and late/run-time binding:

Pure Virtual function:

#### Virtual function puzzles

1. What happens when virtual function is called from the base class member function?

OUTPUT:

EXPLANATION:

1. Whether virtual function will be allowed to call when it is placed under the private access specifier?

OUPUT:

EXPLANATION:

## CLASS AND CLASS RELATED FEATURES

### CONSTRUCTOR & DestruCtor

#### Constructor

Whether default arguments can be used in constructor?

Difference in usage of empty parenthesis

Stack memory allocation

Dynamic Memory Allocation

Converting constructor

Notes

#### COPY CONSTRUCTOR

Note:

#### DeSTRUCTOR

Can we call the destructor manually?

Virtual destructor

#### CONSTRUCTOR & DestruCtor PUZZLES

1. Whether constructor will get called when we declare the object for non-parameterized constructor with empty paranthesis?

OUTPUT:

EXPLANATION:

1. Whether data member of the class can be accessed when object is declared with empty parenthesis?

OUTPUT:

EXPLANATION:

1. Whether conversion constructor will act as a conversion operator?

OUTPUT:

EXPLANATION:

1. Whether conversion constructor will acts as conversion operator when assignment operator is overloaded to its specific type?

OUTPUT:

EXPLANATION:

1. Whether copy constructor will get called when object is returned in the local function?

OUTPUT:

EXPLANATION:

1. Whether default constructor will be created when copy constructor is written explicitly?

OUTPUT:

EXPLANATION:

1. Whether copy constructor can be defined by taking parameter as value rather reference?

OUTPUT:

EXPLANATION:

1. Can we access the private members of object which is received as parameter in public constructor?

OUTPUT:

EXPLANATION:

1. What is difference in calling the constructor with and without parenthesis?

OUTPUT:

EXPLANATION:

### MISCELLANEOUS

#### Difference between class and struct in c++

#### CLASS MISCELLANEOUS PUZZLES

1. Whether public members of base class is accessible when derived one is struct and no access specifier is specified during inheritance?

OUTPUT:

EXPLANATION:

1. Whether non-const function can be called by const object?

OUTPUT:

EXPLANATION:

1. Can we modify the address of this pointer in member function of class?

OUTPUT:

EXPLANATION:

1. Can we change the object which is referred by the this pointer?

OUTPUT:

EXPLANATION:

1. Whether inner class member function can access the private members of outer class?

OUTPUT:

EXPLANATION:

### INHERITANCE

#### INHERITANCE PUZZLES

1. what is the size of the derived class when it inherits the classes which have common base class?

OUTPUT:

EXPLANATION::

1. What happens when we miss the virtual keyword for one class in diamond problem?

OUTPUT:

EXPLANATION:

1. What is size of the empty class which has diamond problem?

OUTPUT:

EXPLANATION:

### MEMBER FUNCTIONS

#### MEMBER FUNCTION PUZZlES

1. Whether member of base class can be accessed in the derived class when base class member functions are overloaded or overrided?

OUTPUT:

EXPLANATION:

### Modifiers

#### MISCELLANOUS MODIFIERS

Final

Member functions

Override

Why override keyword is required?

### OBJECTS

Intialization related features

Intializer list

What is initializer list?

#### OBjects puzzles

1. What is the size of empty class?

OUTPUT:

EXPLANATION:

Rule of five:

Object Slicing

Casting beween objects:

upcast

downcast

Instantiating the inner class in c++

Exception:

RAAI

CADRe

Instantiation:

Classification based on time of instantiation:

early instantiation:

LAZY instantiation:

Wrapper Class

Casting

Static cast

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Copy constructor in c++

**What is copy constructor?**

**When copy constructor is called ?**

**Syntax for copy constructor :**

**What happens if copy constructor is not created by user?**

**Where the copy constructor is must?**

CONSTRUCTOR PUZZLES

1. How order of object construction is decided when more than one class object is kept as member of the class?

OUTPUT:

EXPLANATION:

1. Which constructor is called when object is declared using empty initialize list?

OUTPUT

EXPLANATION

CLASS PUZZLE

1. What happens, when an initializer variable is present in more than one scope and if class member is initialized with such kind of intializer?

OUTPUT:

EXPLANATION:

ENUM PUZZLE:

1. What happens when we use the negative value in the enum value?

OUTPUT:

EXPLANATION:

### EXCEPTION IN C++

need for exception:

try throw catch blocks:

try block:

catch block:

throw block:

throw after the funciton signature:

ellipis operator in catch block:

## STANDARD LIBRARY

### STANDARD TEMPLATE LIBRARY(STL)

#### CONTAINER CLASSIFICATION

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

#### CONTAINER CORRECT USAGE

Factors influencing in choosing correct STL

KEY

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| CONTAINER | FREQUENT INSERTION | FREQUENT DELETE | SEARCH | INSERTION ORDER OF ITEMS | POSSBILE INTERNAL REPRESENTATION | Random access | ADAPTION |
| VECTOR | NR | NR | NR | M | Static Array | Possible | OWN |
| MAP | R | R | R | NM | RB Tree | Not Possible | OWN |
| LIST | R | R | NR | M | Double linked list | **Not Possible** | OWN |
| SET | R | R | R | NM | RB Tree | Not Possible | OWN |
| DEQUEUE | R | NR | NR | M | Array to Array. Master array maintains the list of array. | Not Possible | OWN |
| QUEUE | NR | NR | NR | M | Depends on the Adaptor | Not Possible | Container adaptor |
| ARRAY | NA | NA | NR | M | Array | Possible | OWN |
| FORWARD LIST | R | R | NR | M | Singly linked list | Not possible | OWN |
| STACK | R | NA | NA | M | Depends on the Adaptor | Not possible | Container adaptor |
| UNORDERED MAP | R | R | R | NM | Hashing | Possible | OWN |
| UNORDERED SET | R | R | R | NM | Hashing | Possible | OWN |
| PRIORITY QUEUE | R | NA | NR | NM | Depends on the Adaptor | Not possible | Container adaptor |
| MULTIMAP | R | R | R | NM | Red black tree | Not possible | OWN |
| MULTISET | R | R | R | NM | Red black tree | Not possible | OWN |
| UNORDERED MULTI MAP | R | R | R | NM | Hashing | Possible | OWN |
| UNORDERED MULTISET | R | R | R | NM | Hashing | Possible | OWN |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| CONTAINER | Deleting any element in container | Position based Access |  |  | |  |
| VECTOR | Possible | Possible |  | |  |  |
| MAP | Possible | Not possible |  | |  |  |
| LIST | Possible | Not possible |  | |  |  |
| SET | Possible | Not possible |  | |  |  |
| DEQUEUE | Possible | Possible |  | |  |  |
| QUEUE | Not possible | Not possible |  | |  |  |
| ARRAY | Not possible | Possible |  | |  |  |
| FORWARD LIST | Possible | Not possible |  | |  |  |
| STACK | Not possible | Not possible |  | |  |  |
| UNORDERED MAP | Possible | Possible |  | |  |  |
| UNORDERED SET | Possible | Possible |  | |  |  |
| PRIORITY QUEUE | Not possible | Not possible |  | |  |  |
| MULTIMAP | Possible | Not possible |  | |  |  |
| MULTISET | Possible | Not possible |  | |  |  |
| UNORDERED MULTI MAP | Possible | Possible |  | |  |  |
| UNORDERED MULTISET | Possible | Possible |  | |  |  |

#### TIME AND SPACE COMPLEXITY of STL

|  |  |  |  |
| --- | --- | --- | --- |
| CONTAINER | INSERT | DELETE | SEARCH |
| Array | O(1) | NA | O(N) |
| VECTOR | O(1) for back and O(N) for others | O(1) for remove  O(n) for erase | O(N) |
| MAP | O(log N) | O(log N) | O(log N) |
| LIST | O(N) | O(N) | O(N) |
| SET | O(log N) | O(log N) | O(log N) |
| DEQUEUE | O(1) | O(1) | O(N) |
| QUEUE | O(1) | O(1) | O(N) |
| FORWARD LIST | O(1) | O(N) | O(N) |
| STACK | O(1) | O(1) | NA |
| UNORDERED MAP | O(1) | O(1) | O(1) |
| UNORDERED SET | O(1) | O(1) | O(1) |
| PRIORITY QUEUE | O( log N) | O(log N) | O(N) |
| MULTIMAP | O(log N) | O(log N) | O(log N) |
| MULTISET | O(log N) | O(log N) | O(log N) |
| UNORDERED MULTI MAP | O(1) | O(1) | O(1) |
| UNORDERED MULTISET | O(1) | O(1) | O(1) |

### Terminology

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |

#### GENERAL STL INTERVIEW QUESTIONS

1. How capacity of the standard template library is incremented?

OUTPUT:

EXPLANATION:

### UTility functions

Functor

Lambda function

Syntax:

Accessing all local variable inside lamba function

ACCESSING ALL LOCAL VARIABLES AS REFERENCE

ACCESSING ALL LOCAL VARIABLES AS VALUE

calling the lambda function:

for\_each function

Lambda function for for each:

### map in c++

INTERNAL PROCESSING OF MAP:

PROPERTIES OF MAP:

SYNTAX:

Eg:

ACCESSING CONTENT OF MAP:

ARRAY LIKE ACCESS:

INSERTING CONTENT TO MAP:

UTILITY FUNCTIONS IN MAP:

FIND SIZE OF MAP:

Insert\_or\_assign()

### VECTOR

#### Vector puzzles

### ITERATOR IN C++

Arithemetic operations on iterator:

SET

### BIT FIEld

#### BIT FIELD PUZZLES

1. What is the size of class when bitfield is used in the class?

OUTPUT:

EXPLANATION:

1. What happens when arithemetic overflow occurs for bit field?

OUTPUT:

### UTILITY FUNCTIONS

#### fill function in c++

SYNTAX:

SAMPLE PROGRAM:

LIMITATIONS ON FILL FUNCTION:

PROGRAMMING USAGE:

## TEMPLATE

DEFINING THE TEMPLATE CLASS:

DIFFERENCE BETWEEN CLASS AND TYPENAME IN TEMPLATE:

CREATING OBJECT FOR TEMPLATE CLASS:

CREATING TEMPLATE FUNCTION INSIDE THE TEMPLATE CLASS:

TYPE CASTING OF FUNCTION TEMPLATE:

SAMPLE PROGRAM:

OUTPUT:

How template function or class is works internally?

### UTILITY

is\_same template class:

#### UTILITY PUZZLES

1. Whether objects can be used to compare the two things in is\_same template class?

OUTPUT:

EXPLANATION:

### YET TO CATEGORIZE

Impact on function overloading:

Class template:

function template:

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |

## INPUT AND OUTPUT

Manipulator function:

std::endl

Static function

#### CONSOLE INPUT AND OUTPUT

#### FORMATTING

#### Streams

Basic\_iostream

Naming convention

### FILE HANDLING IN C++

Reading file in c++

Initializing object for file:

Opening a file manually:

|  |  |
| --- | --- |
| class | default mode |
| ofstream | ios::out |
| ifstream | ios::in |
| fstream | ios::in | ios::out |
| Mode | Description |
| ios::in | Performs input operation |
| ios::out | Performs output operation |
| ios::app | All output operation performed at the end of the file. This indicates the rest of file is remain untouched, |
| ios::binary | Binary mode |
| ios:trunk | Deletes the content of the old file and replace with the new one. |
| ios:ate | Set the cursor at the end of the file |

Read a file using a overloaded operator <<

getline() function

Read() method of istream:

Manipulation on file position:

clear() method of istream:

Number of characters read gcount:

Ignore the content up to some delimiter:

Write a content to file:

## C++ THIRD PARTY LIBRARY

### ROGUEWAVE LIBRARY

#### RWUString

Toutf8()

#### Dates

RWDBDATETIME

### TYPES

#### String

String Literals

Multibyte character

### RW-JOINS

### Miscellaneous

From()

## C++ Miscellaneous

### Difference between C and C++: More than you know

USE OF elaborated type specifier:

CALL BY REFERNCE:

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

### Difference between c++11 and below version

### CORE DUMPED (SEGMENTATION FAULT)

1.changing and accessing the value of the null pointer

using throw statement for undefined catch block

Using %s for int in pointer:

Changing the content of char pointer

Unlimited recursive calls:

Dividing 1 by Zero

### The code secrets you may not know

intializing a variable

INITIALIZING VARIABLE IN PARAMTER

### C AND CPP DIFFERENCES

1. INTIALIZING POINTER WITH VALUE WITHOUT TYPECASTING:
2. Operator overloading and function overloading is supported in c++ and not in c
3. Multiple declaration of global variables is allowed in c
4. Using a symbol(function) without declaring
5. void \* is implicitly converted any type of pointer in c but not in c++

### COMPARISON BETWEEN C++98 AND C++11

STRUCT TM :

### GOOD PROGRAMMING PRACTICE

### WEIRD FACTS AND INTERVIEW QUESTIONS

1. Why c++ does not support the unsigned floating point:
2. Can we declaring variables at any places in the block:
3. What is dual in “select \* from dual” ?
4. Different types of NULL usage and their meaning in program
5. Write the program to print the executable name
6. what function will return when function with no return statement and return value is specified for function?
7. whether sizeof of two different pointers will vary in same machine?
8. Opaque pointer and opaque data type

Opaque data type:

Information hiding

Transparent data type

Declaring local object for the opaque data type

Calculating the size of opaque data type

Advantages of opaque pointer

Disadvantages of opaque pointer

Application of opaque pointer:

1. How to achieve the data encapsulation in c?
2. Why C++ standard not recommending to initialize the variable to zero ?
3. Why following program allows to compile when function prototype or implementation of function is not present before calling that function?
4. Program to print a string without main function:
5. Multiple declaration of global variable with same variable name
6. How to allocate the dynamic char array without using char pointer in struct?
7. What is the size of the class when char array is declared with size 0?
8. Can we declare a array with empty size is class or struct?
9. Whether Calling a function without function declaration in C or C++ is allowed?
10. Why array index in C and C++ was starting with zero(0) instead of one(1)?
11. Whether any performanace impacts will occur when we declare variable inside and outside of for loop in c or c++ programming?

Why there is no performance impacts for POD Types:

What about non-POD types like class?

1. Whether 0(zero) is decimal literal or octal literal in C or C++ programming?
2. What is difference between user and schema in oracle database?
3. What is difference between const char \*ptr and char const \*ptr in C programming?
4. Can we call the non-volatile member function using the volatile object?

Answer:

Simple Answer:

1. Whether const and volatile keyword is considered in the function overloading?

O/P:

Explanation:

1. When destructor of the local objects will be called when go to is used within the function block?

Answer:

Explantion:

1. Whether non-type template parameter can be assigned with some value?

Output:

Explanation:

1. What happens when extern “C” is used without block in the program?

Output:

Explanation:

1. What happens when float argument is passed to overloaded function with int and char as parameters(Type Conversion rules)?

Output:

EXPLANATION

1. Whether user defined objects is allowed as non-type template parameter?

Output:

EXPLANATION:

1. Priority of implicit conversion of user defined type and system defined type

Output:

EXPLANATION:

1. When constructor and destructor will be called for various object declaration methods in C++ classes?

Output:

EXPLANATION:

### include files

#include”” and #include<>

#include”myheader.h”

#include<stdlib.h>

Similar confusing definitions:

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

other noting points:

### Cpp applications

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

### Undefined behavior in Programming

WHY UNDEFINED BEHAVIOR:

DIFFERENCE BETWEEN UNSPECIFIED AND UNDEFINED BEHAVIOUR:

HOW TO OVERCOME UNDEFINED BEHAVIOUR IN PROGRAMMING:

UNDEFINED BEHAVIOUR IN C AND C++:

USING THE VARIABLE WITHOUT INITIALIZATION:

FUNCTION WITHOUT A RETURN STATEMENT:

VARIABLE SIZE OVERFLOW

DEREFERENCING A DANGLING POINTER

DEREFERENCING THE WILD POINTER:

RETURNING A ADDRESS OF LOCAL VARIABLE

CHANGING THE CONST CHAR POINTER

### COMPILATION AND LINKING

Compiler for c++

clang

gcc

Header files

Precompiler header files:

File extenstion

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

### C++11

New features of C++11

### C++14

New features of C++14

### HEADER FILES FOR COMMONLY USED FUNCTIONS

|  |  |  |
| --- | --- | --- |
| Functions | Header File | |
| Sleep | Unistd.h | |
| Strcpy | Cstring | |
| Read,write,close | Unistd.h | |
| Sstream | Iomanip | |
|  |  | |
|  |  | |
|  |  | |
|  |  | |
|  |  | |
|  |  | |
|  |  | |
|  |  | |
|  | |  |

### MATH OPERATIONS

1. abs() gives the absolute value.

### STANDARDIZATION

Terminolgy:

Defacto standard

ISO STANDARD

ISO Standard releases

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

### RETURN VALUE OPTIMIZATION (RVO) AND COPY ELLISION

Before learning the Return value optimization. Just understand the following concepts

As-If rule:

SIDE EFFECTS:

### OOPS CONCEPTS IN C++

#### Abstraction

Class which contain pure virtual function is called abstraction.

### TRAINING

#### Practice problems

Grading Program

* 1. Write a program that allows the user to enter the grade scored in a programming class (0-100).
  2. If the user scored a 100 then notify the user that they got a perfect score.
  3. Modify the program so that if the user scored a 90-100 it informs the user that they scored an A
  4. Modify the program so that it will notify the user of their letter grade 0-59 F 60-69 D 70-79 C 80-89 B 90-100 A

Employee Information

Focusing class related concepts

* 1. Create employee class
  2. Keep employee info in members
  3. Print employee info when requested
  4. Create BnyMellon Employee class and derive from employee
  5. Keep employee id in private
  6. get employee information from user
  7. Get 5 employee information and store it.
  8. Find employee by employee id and print their details

### COMPETITIVE PROGRAMMING – PREPARATION PROGRAM

Reading a line

Parsing a string

Convert string to integer

Integer to string

Key Points to remember

PROGRAM

# C PROGRAMMING

## TYPES

### initialization concepts:

intialization of external variables:

intialization of variables using ocatal number:

intialization of two variables at a time:

intialization of array using fill :

Intializing the non-static class members in declaration

### VARIABLE ARGUMENTS IN C

### Sizeof and Sizeof():

#### CASTING

Functional casting

C-Style casting

### POINTERS

#### 

#### confusing pointer definitions

1.null pointer:

2.void null pointer:

3const pointer and char constant:

4.Dangling pointer

5.raw pointer

### EXTENDED DATA TYPE

## PREPROCESSING DIRECTIVES

### DIRECTIVE

DIRECTIVE:

#line directive

### MACRO

STRINGNIFY OPERATION IN MACRO

Properties:

#### MACRO PUZZLES

1. Difference between # and ## in macro expansion?

OUTPUT:

EXPLANATION:

1. Whether stringnification operation will be converted to string literal?

OUTPUT:

EXPLANATION:

1. Whether macros used after # will expand further?

OUTPUT:

EXPLANATION:

1. Whether macros can be defined within local scope?

OUTPUT:

EXPLANATION:

## STRUCTURE

### Tag name in struct

WHY TYPEDEF IS GIVEN OFTEN IN STRUCT:

NORMAL IDENTIFIERS AND TAG NAMES:

#### STRUCT PUZZLES

1. Whether object name can be same as class name?

OUTPUT:

EXPLANATION:

## STANDARD LIBRARY FUNCTION

### UTILITY FUNCTION

#### Perror and strerr

perror:

strerr:

SAMPLE PROGRAM:

### TIME

C provides the standard library to calculate the system time. In c you need to include the header file time.h and in c++ you need to include <ctype>

DATE\_TYPE:

Note on epoch time:

struct tm:

FIND THE TIME DIFFERENCE:

time function:

localtime function:

mktime function:

asctime function:

strftime

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

gmtime:

ctime:

Clock function:

Sample Program:

Miscellaneous information

Mathematical Related Utility

strtol

### STRING IN C

## LIBRARY

#### DYNAMIC LIBRARY

Import Library

## INPUT AND OUTPUT OPERATIONS

### FILE HANDLING

System calls involved in file

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

#### Reading the input from stream

gets:

puts:

fgets:

fputs:

Scanf :

Excluding of characters

# C and C++ Generic Concepts

## building the application

TERMINOLOGY:

TRANSLATION UNIT

What is intermediate form?

PREPROCESSING:

## ERROR AND WARNINGS AT RUN TIME AND COMPILE TIME

1. BUS ERROR

# DESIGN

## OBJECT ORIENTEd CONCEPTS (OOPS)

List some oops concepts:

Terminology

object:

Class

Object Oriented Programming (OOP)

Difference between object oriented and object based language?

Object oriented:

Object Based language:

#### Abstraction

Interface

When to use interface?

Abstract class

Need for abstract class

When to use abstract class?

Difference between abstract class and interface?

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

#### ABSTRACTION

Existing in idea, but no physical existence.

Metaphor

#### POlymorphism

Metaphor

#### INHERITANCE

1. Used to get the properties of the any existing class and adding enhancing the existing one.

Metaphor

Types:

Using Inheritance scope of job is high

#### ABSTRACTION

Existing in idea, but no physical existence.

Metaphor

#### POlymorphism

Metaphor

#### INHERITANCE

Metaphor

Types:

Using Inheritance scope of job is high

### UNCATEGORIZED – PERFORMANCE

1. Master/Slave port

### DATA MANAGEMENT

#### BRINGING DATA TO APPLICATIOn

Huge columns and column will be picked by data provider

#### PERFORAMANCE TUNING- LARGE DATA

1. Partitioning of data
2. Virtualization
3. Multithreading
4. Asynchronous processing

### MICROSERVICE

#### cpp Microservice

# PROGRAMMING

## Static analyzer

CWE – Common vulnerability enumeration

MISRA

## COVERITY

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## DEBUGGING & BUILD MANAGEMENT

#### 

#### PROFILING

List of profiling tools

Gprof

#### GDB DEBUGGER

Compilation of program to use debugger:

# THEORITICAL COMPUTER SCIENCE

## TERMINOLOGY

FORMAL SYSTEM

Web service:

Interoperable:

## PROGRAMMING LANGUAGES

### TERMINOLOGY

Programming paradigm:

Strongly typed and weakly typed

Strongly typed language

weakly typed language

Context switching

Static and dynamic language

Static language

Dynamic Language:

### Differences

Difference between script and programming language

## software engineering

TERMINOLOGY:

DEPRECATED:

OBSELETE

## type theory

## GRAPH THEORY

Definition:

Classification of graph:

1. Based on relationship between nodes

Terminologies

|  |  |
| --- | --- |
| TERMINOLGY | DESCRIPTION |
| Acyclic graph | Graph with no cycles is called as acyclic graph |
| Mixed graph | Graph that contains directed and undirected edges is called as mixed graph |
| Multiedges | When same two vertices is connected by multiple edges then it is called as multi edges |
| Multigraph | It is an undirected graph which contains the multiedges |
| Loop | When edge refer to same vertex, it is called as loop |
| Simple graph | Undirected graph with no loop and multiple edges |
| Quiver (or) Directed Multigraph | As name suggests, it is directed multigraph |
| Weighted graph | Weight or cost is attached to each edge in the graph |
| Half edges, no ends, loose edge | edges of the graph is connected to only one of the vertex and other end of the vertex is simply left |
| Regular graph | Every vertex has same degree |
| Complete graph | Each pair of vertices is connected by edge |
| Finite graph | When vertex and edge of the graph is finite it is called finite graph |
| Oriented graph | Directed Graph |
| Connected graph | **Connected graph in Undirected graph:**  when we take any two vertices in the graph and if there is a connection between two vertices in the graph, then it is called connected graph.  **Connected graph in directed graph:**  There are two types   1. Strongly connected graph 2. Weakly connected graph   For any ordered pair in the graph, if there is directed connectivity between them then it is called strongly connected graph.  For any ordered pair in the graph, if there is no directed path between two vertices, and if connectivity happens when directed path in the graph is replaced with undirected path, then it is called weakly connected graph. |
| Bipartite graph | “Bi”  Entire graph is divided in two groups  “Partite”  Partitioning  In bipartite graph, no two vertices in a same group will be connected by a same edge. |
| Complete Bipartite graph | When every vertex of one group is adjacent to every vertex of other group it is called as complete bipartite graph. |
| Di graph | It is also called as directed graph |

Circuit,walk,path,cycle,trail

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Terminology | Vertices | Edges | CLOSED/OPENED | Mind Map |
| Path | Don’t repeat | Don’t Repeat | open | Consider straight road with no signal from one destination to other. |
| walk | Repeat | Don’t Repeat | Closed or open | Consider straight road with signal from one destination to other. So that they divert to reach the same node again to avoid traffic. |
| Circuit | Repeat | Don’t Repeat | Closed | Same as walk but at end we will be reaching the same destination where we started. |
| Trail | Repeat | Don’t Repeat | Open | Same as walk but at end we will not be reaching the same destination where we started. |
| Cycle | Don’t Repeat | Don’t Repeat | Closed | Same as path, but reaching the same destination where we reach. |

Graph operations

Unary Operation:

Binary Operation

Graph Minor

Graph Traversal

Depth first traversal:

Depth first search for disconnected graph:

Tree Vs Graph

Breath first search

Applications of graph

## NUMBER sYSTEMS

Generic concepts:

Binary Number system

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

## BANKERS ALOGORITHM - oS

## DATA AND INFORMATION

Ascii:

# DATABASE

#### JOINS

Using more condition in join’s on

What’s the difference

### SQL SERVER

#### Types

Wide characters

#### UNCATEGORIED

1. XML data can be directly used in SQL query.

### DATABASE COMPARISON

|  |  |  |
| --- | --- | --- |
| NAME | SQL SERVER | ORACLE |
| All table information |  | Sys.tables |
| Change case | upper() | upper() |
| Date | Currentdate() | sysdate |
|  |  |  |

## SQL SERVER DEVELOPER

### SHORTCUT KEY

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |

## PL/SQL DEVELOPER

### SHORTCUT KEY

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |

## ORACLE PRO\*C/C++

## ORACLE SQL

### QUERY PUZZLES

Second maximum salary in table

### VIEW

Advantages

Disadvantages

Types – Based on Data Storage

Materialized View

## ORACLE-Pl/SQL

### CONCEPTS

Advantages of PL/SQL

### STRING MANIPULATION

Find a string in a string

Get a substring in string

### Concepts

Handling NULL RELATED FUNCTIONS

1. Is null
2. Decode
3. Coalesce
4. Nullif

Difference between sql and PL/SQL

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

#### INDEX

Types of index

#### TRIGGER

Types of trigger

#### CURSOR

Advantages

Disadvantages

Difference between function and procedure

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## ORACLE-SQL

### Concepts

Difference between truncate and delete

COUNT

What is difference between count(\*) and count (1)?

Data Integrity

### STANDARD PL/SQL LIBRARY

#### FILE READING

## COMMON THERORITICAL CONCEPTS

### TERMINOLOGY

RAC

Execution plan

Difference between Case and decode

### PERFORMANCE

Ways to improve performance when load increases

#### Keys in Database

Super key

Candidate key

Primary Key

#### Database Normalization

1 st Normal form

2nd Normal form

3rd Normal form

## USING PARTITION CLAUSE FOR AGGREGATE FUNCTIONS

# WINDOWS BASED PROGRAMMING

### BUILD TOOLS

Dumbbin

Lib.exe

### LIBRARY

#### STATIC LIBRARY

#### DYNAMIC LIBRARY

Import library

#### STATIC AND DYNAMIC LIBRARY – COMMON DETAILS

Lib extension files

### Multithreading

SampleCode

### COMMANDS

#### Find

|  |  |
| --- | --- |
| “\*+\*” | To search for + then, “\*+\*” |
| File: | To search only file |
| Content: | To search in file contents |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# UNIX AND UNIX RELATED PROGRAMMING

## MULTITHREADING-UNIX

### PTHREAD LIBRARY

Creating a thread:

PROTOTYPE:

EXAMPLE:

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

passing paramter as null:

Default attributes to thread if attribute is not defined

synchronization in threads:

Thread Attributes:

What is the scope of the attribute objects?

Helper functions to set the thread attribute:

Pthread\_attr\_init()

pthread\_attr\_setdetachstate

pthread\_attr\_getdetachstate

pthread\_join

pthread\_attr\_destroy:

Advantages of giving attribute for thread creation:

THREAD POOL:

### Miscellaneous Concept about threads

#### 

#### Difference between synchronization and serialization:

#### Spurious wakeup in threads:

pthread condition for wait and signal

#### Difference between mutex, locks, semaphore

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

#### Difference between BINARY semaphore & MUTEX

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |

#### Maximum thread allowed per process

#### WHAT HAPPENS WHEN PARENT THREAD IS DIED BEFORE CHILD THREAD IS ALIVE?

TASKS

## SOCKET PROGRAMMING

### BSD Socket

sys/socket.h:

Ports used for sending:

Protocol Family:

File Descriptor:

Compilation:

Address family and Protocol family:

#### Functions and their usage

Socket();

bind();

|  |  |
| --- | --- |
|  | |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

LISTEN FUNCTION:

|  |  |
| --- | --- |
|  | |
|  |  |
|  |  |
|  |  |
|  |  |

ACCEPT FUNCTION:

|  |  |
| --- | --- |
|  | |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

HELPER FUNCTIONS:

recv, recvfrom, recvmsg:

send , sendto, sendmsg

Read and write:

Diff between read/write and send/recv:

Miscellaneous Information:

SELECT OR POLL:

|  |  |
| --- | --- |
|  | |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Helper macros that is used for select function:

Poll method:

get socket options

|  |  |
| --- | --- |
|  | |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Difference between sockaddr and sockaddr\_in

UTILITY FUNCTIONS:

|  |  |
| --- | --- |
|  | |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

close();

### MISCELLANEOUS

Major program difference overview for TCP and UDP

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Blocking system calls

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

System calls used in TCP and UDP

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Functions which return FD

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Difference between TCP and UDP

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |

When to use TCP Over UDP?

Real time application of TCP and UDP

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

### TCP IMPLEMENTATION

#### Server Program

#### CLIENT PROGRAM

### UDP PROGRAM

#### SERVER PROGRAM

#### CLIENT PROGRAM

## SIGNALS

Error which occur in signals

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |

## HARD AND SOFT LIMIT IN UNIX

Hard Limit

Soft Limit

## STANDARdS

## SEMAPHORE

How semaphore is used for producer consumer problem?

Snapshot of function calls

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

Implementation of semaphore server

Implementation of semaphore client

## MISCELLENEOUS

### System RESOURCES

#### SYSTEM FILES

.profile:

# PRO \*C PROGRAMMING

## TERMINOLOGY

OCI

## DYNAMIC SQL STATEMENT IN PRO \*C/C++

LIFE CYCLE OF DYNAMIC SQL STATEMENT:

COMPOSING THE DYNAMIC SQL STATEMENT:

Place holder for host variable:

## More about varchar and sql\_context

## Basics of PRO C

After Precompilation:

## Connecting with database

DESCRIBTION OF THE PROGRAM:

other forms of connect statement:

connecting to database:

advanced connection technique:

## Datatypes in Pro\*C/C++

Internal Datatype:

long:

varchar2 & varchar

FLOAT DATATYPE

Varibles in Pro \*C/C++

Host Variable:

Indicator variable:

Host array:

## SQL STATEMENTS IN PROC

Declare statement in sql:

DELIMITERS:

## DATABASE CONCEPTS USED IN PRO \*C/C++

COMmIT:

SET TRANSACTION

explicit lock:

handling char data:

|  |  |  |
| --- | --- | --- |
| Option | Blank Padded | Null terminated |
| varchar | yes |  |
| string |  | yes |
| charz | yes | yes |
| charf | yes |  |

inline precompiler option:

Naming Files:

Placing inside the program:

Comments lines:

datatype equivalencing:

shared & Private sql areas:

Cursor

transaction:

Line continuation:

maxliterals:

conditional precompilation:

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

CUrsor in modular programming

## CURSOR

declaring the cursor

open the cursor:

## context

CREATING CONTEXT:

ALLOCATE THE CONTEXT:

USE THE CONTEXT:

DEFAULT IN CONTEXT:

OUTPUT:

## Pointers & Memory management

### new operator in c++

creating the object without new:

creating the pointer to class:

invalid use of new operator:

difference between giving class name and funtion after new operator:

Using new operator for basic datatype:

heap memory and static memory:

## Make FILE

Running a make file

Command Line in Make:

Targets and prerequisite:

Rule in make:

Dependency lines:

SHELL lines:

MACROS:

Using shell command in make file

Automatic variables:

PHONY Targets:

## Miscellaneous

## Unexplored c areas

## Header Files

### Conditional compilation:

|  |  |
| --- | --- |
| Condition | Description |
| #Ifdef | If this macro is defined |
| #Ifndef | If this macro is not defined |
| #If | Test if a compile time condition is true |
| #Else | The alternative for #if |
| #Elif | #else an #if in one statement |
| #Endif | End preprocessor conditional |

# UNIX CONCEPTS

## How to check the port was listening(UNIX)?

## MISCELLANEOUS

### library files in unix

static library:

Dynamic library:

Difference between static and Dynamic(Shared) Library

library naming convention:

#### Sample Program to access Library

## UNIX Commands

### NETWORK REALTED COMMANDS

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Command to get the process id of given port number

### Grep command in UNIX

Extracting a piece of text in file:

Extracting a piece of text in directory:

Extracting a piece of text in same kind files:

Extracting a piece of text in multiple specified files:

Extract all except the given pattern:

Extract the words which contains the spaces:

Stroing the grep pattern to file:

case-insensitive search:

Match the whole words only:

Count the number of lines that matched with grep:

List the files names for which the grep pattern matches

Matching with starting word of line:

Matching with ending word of line:

Extracting the line contains the given word

search for blank line in grep

searching the special character

Matching any one of the given character

Ignoring the match of given character set

Match with prefix or suffix

Grep that excludes some pattern

variants of grep:

|  |  |
| --- | --- |
| OPTION | Description |
| h | Suppress the file name |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

SAND BOX:

### UNIX COMMAND LINE MEANING

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| A |  |  |  |
| B |  |  |  |
| C |  |  |  |
| D |  |  |  |
| E |  |  |  |
| F |  |  |  |
| G |  |  |  |
| H |  |  |  |
| I |  |  |  |
| J |  |  |  |
| K |  |  |  |
| L |  |  |  |
| M |  |  |  |
| N |  |  |  |
| O |  |  |  |
| P |  |  |  |
| Q |  |  |  |
| R |  |  |  |
| s |  |  |  |
| t |  |  |  |
| u |  |  |  |
| v |  |  |  |
| w |  |  |  |
| x |  |  |  |
| y |  |  |  |
| z |  |  |  |

MISCELLANEOUS COMMANDS

|  |  |
| --- | --- |
| Command | Description |
| ssh user@ip |  |
| uname |  |
| uname –a |  |
| id |  |
| du –sh \* | used for finding the disk usage  -h for human readable format |
| ldd executablename | Shows where all the library fiels are linked |
| pkg-config |  |
| cpp -v |  |
| truss | This is usually used for debugging the running application.  Listing the system call made by the executable at certain point of time. Unfortunately this is not part of POSIX Single unix specification |
| shutdown | To shutdown the unix system |
| gcore | To generate the core file based on the process id. |
| cut | cut –d delimiter, cut –f field, cut –c range |
| gcc –version | Shows the version details of GCC |
| sort | Sort the given input |
| uniq | To remove the repeated lines. |

List Command

|  |  |
| --- | --- |
| Command | Description |
| ls –1 | One by one |
| ls –R | Includes Subdirectory |
| ls -lrt |  |
|  |  |

substitution Command

|  |  |
| --- | --- |
| Command | Description |
| :1,100s/find/replace/g | finds and replace the word (all occurances) |
| :1,100s/find/replace/gc | **finds and replace the word (confirm each time)** |
| :%s/find/replace/gc | Find and replace in entire file |
|  |  |

Miscellaneous Command

|  |  |
| --- | --- |
| Command | Description |
| passwd | Changes PWD |
|  |  |
|  |  |
|  |  |

FIND COMMAND

|  |  |
| --- | --- |
| Command | Description |
| find curdir –name “filename” | finds file with case sensitivity |
| find curdir –name “filename”` | finds file with w/o case sensitivity |
| find . /home/dir1 /home/files –name “filename” | Search all listed directory |
|  |  |

EditOr command

|  |  |
| --- | --- |
| Command | Description |
| vi filename | vi editor |
| vi file1 file2 | Open Multiple files |
| ed file | Line editor |
|  |  |

Changing directory

|  |  |
| --- | --- |
| Command | Description |
| cd ~ | Return to login directory |
| cd / | Returns to the Entire system root directory |
| cd ~username | Takes to the other user login directory if permission given |
|  |  |

Process status

|  |  |
| --- | --- |
| Command | Description |
| ps | process status |
| ps –e | List the process that is running now |
| ps –f | Generates the full listing |
|  |  |

Copy command

|  |  |
| --- | --- |
| Command | Description |
| cp –r dir1 dir2 | copy one directory with other |

Move command

|  |  |
| --- | --- |
| Command | Description |
| mv ../Current directory/\* target directory | Move all files from source director |

Delete commands

|  |  |
| --- | --- |
| Command | Description |
| rm –r dirname | Removes the directory |

## Cron tab

cron tab:

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# APTITUDE

## QUANTITATIVE APTITUDE

### TRAIN PROBLEM

#### TERMINOLOGY

RELATIVE SPEED

Same direction

Opposite direction

#### FORMULA

# TELECOM AND NETWORKING

## socketS - NETWORKING

Socket address:

DIFFERENCE BETWEEN IP ADDRESS AND PORT (REALTIME EXAMPLE 1)

DIFFERENCE BETWEEN IP ADDRESS AND PORT (REAL TIME EXMAPLE 2):

API:

## STANDARDS

### OSI LAYER

1. Application layer

Protocols:

Example:

1. Presentation layer

Protocols:

Example

1. Session layer

Protocol

Example

1. Transport layer

Protocols:

Example:

1. Network layer (packet)

Protocols

1. Data link layer (frame by frame delivery)

Protocol:

1. Physical Layer (bit by bit delivery)

Protocol

Example

## Local host

MECHANISM BEHIND LOCALHOST:

NEED FOR LOCALHOST:

CAN WE USE LOCAL HOST WITHOUT NETWORK?

## BASICS

### TERMINOLOGY

URI

SSID

NIC

Packet & circuit switching

### COMMANDS

Ping

Telnet

### IP ADDRESSING

PROPERTIES

### WIRELESS LAN

TERMINOLOGY:

AP

BSS

STANDARD

## NETWORK PROTOCOLS

### TCp/IP

When to use TCP over UDP?

### MISC PROTOCOLS

#### SOAP

# UNCATEGORIZED TOPIC

## VISUAL STUDIO

### SHORTCUT KEY

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |

### TFS

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

### VISUAL STUDIO DEBUGGER

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

### PERSONAL DEVELOPMENT

#### BEST HABITS

1. Judging against best

### MULTILINGUAL

#### Codes

Unicode

#### ENCODING SCHEME

UTF-8

#### ETC

Why text is scrambled?

### COMMON OPERATING SYSTEM

#### TERMINOLOGY USAGE

|  |  |
| --- | --- |
| WINDOWS | LINUX |
| Command line switches | Command line options |

### BUSINESS INTELLIGENCE

### INVESTMENT BANKING

#### TERMINOLOGY

OLAP

#### SECURITY

Derivative

### OFFICE TOOLS

#### POWERPOINT

### POWERPOINT

Create Powerpoint from text file

### PROGRAMMING – PROBLEM SOLVING

#### CATEGORY

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

#### FREQUENTLY DOING MISTAKES

### BUSINEsS SCHOOL

#### STARTING BUSiNESS

CHECKLIST

### PROJECT BASED QUESTIONS

Explain your current project

Big picture of application

Specialized area in big picture

Technical Part

Role and challenge

What are some challenging things you faced in your project?

### HR QUESTIONS

Tell me about yourself

Personal Information

Impressions

Leadership

Any questions need to be asked?

Why should I hire you?

Why you are leaving the current company?

Why your current is not providing opportunity to enhance your professional skill set?

Will you leave us when you want to broaden your skills further?

### GIT

bisect

### OnlINE PROGRAMMING TESTS

Attitude towards target

Macros

Tokenizing the string

Getting input from user

## FILE EXTENSION AND MEANING

|  |  |
| --- | --- |
|  |  |
| .zip | compressed archive file |
| .rar | compressed archive file |
| .mpeg | motion picture expert group |
| .avi | audio and video interleave, developed by Microsoft in 1992 |
| .art | clip art files |
| .bak | backup files |
| .cbl | cobol code |
| .doc | document file for word |
| .gif | graphics interchange format developed by compuserve for web |
| .htm | hypertext markup |
| .mp3 | mpeg audio layer 3 |
| .pdf | portable document file by adobe |
| .xml | extensible markup language |
| .3gp | 3rd generation partner ship project developed for 3g phones |

## ABBREVATION OF COMPUTER TECHNOLOGY TERMS

|  |  |
| --- | --- |
|  |  |
| ARPA | ADVANCED RESEARCH PROJECT AGENCY (FIRST INTERNET) |
| MICR | MAGNECTIC INK CHARACTER RECOGNISATION |
| PDA | PORTABLE DIGITAL ASSITANCE |
| IP | INTERNET PROTOCOL |
| TCP | TRANSFER CONTROL PROTOCOL |
| .EDU | EDUCTIONAL INSTITUTIONS |
| .COM | COMMERCIL ENTITY |
| .ORG | NON PROFIT ORGANISTAION |
| .NET | NETWORK BASED ORGANISTION |
| TLD | TOP LEVEL DOMAIN(.EDU,.NET.ORG) |
| HTML | HYPER TEXT MARKUP LANGUAGE |
| URL | UNIFORM RESOURCE LOCATOR |
| HTTP | HYPER TEXT TRANSFER PROTOCOL |
| ARPA | ADVANCED RESEARCH PROJECT AGENCY (FIRST INTERNET) |
| MICR | MAGNECTIC INK CHARACTER RECOGNISATION |
| PDA | PORTABLE DIGITAL ASSITANCE |
| IP | INTERNET PROTOCOL |
| TCP | TRANSFER CONTROL PROTOCOL |
| OOAD | Object oriented analysis and design |
| SOA | Service Oriented Architecture |
| WDSL | Web services definition Language |
| URI | Uniform resource identifier |
| URL | Uniform resource locator |
| URN | Uniform resource name |
|  |  |

## VIM EDITOR

Abbrevation of VI editor is Visual Editor. VIM is the improved version of VIM. VI editor is the very old editor which is very difficult to use. VIM gets its popularity by the keyboard shortcuts. We can use the vim without using the mouse that the added advantage.

VIM editor is written by Bram Moolenaar.

It is used in the UNIX like systems .

It is the free and open source software.

Latest version of vim is 7.4 released in Aug 10 2013.

The logo of VIM editor is

navigation:

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

search:

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

editor commands:

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

Customize commands:

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

miscellaneous commands:

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |

## ABBREVIATION TERMS

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |

## PRIME NUMBER PROGRAM FOR LARGE NUMBERS

function inside the loops:

assigning variable inside the loop:

## PUZZLES

## DROP BOX

1. olding,
2. Calculating the performance measurement.

### PROBLEM SOLVING

#### Concepts

#### FREQUENTLY FAILURES

### ROGUEWAVE

#### Criteria

### Cross Platform Development concepts

#### Library

Windows Specific Implementation

#import directive

#### Database

Length of string

Oracle

Sql server

Limiting the number of rows

Sql server

Oracle

Fetch Offset

Limit the rows

Using row number

Data types

ANSI Supported types

List of types

VARCHAR

Oracle Specific types

List of types

CLOB

ODBC

Table Creation

Using existing table

SQL

ORACLE

Type conversion

String to other types

SQL

ORACLE

Column selection

Common table expression

Performance tuning

1. Using exists and in clause

#### UNIX COMMANDS

|  |  |
| --- | --- |
|  |  |
| dirname | 1. Returns the directory name by removing the file name 2. When file name is not specified, the it removes the last most directory |
|  |  |

#### C PROGRAMMING

Enum notes

Numeric limits – Mathematical representation

Unsigned Integer – 0 to (2^n)-1

Signed integer : -(2^n-1) to (2^n-1)-1

Unsigned integer

Notes

It is not recommended to use to save one bit.

UNCATEGORIZED

STRING LITERAL

Memory Layout

1. The place where string literal is physically stored will depend on the operating system
2. String literal will be stored in text segment or data segment.

#### PROCESSOR

X86

It represents that it belongs to INTEL 8086 architecture.

Memory address register

#### ASSEMBLY LANGUAGE

Instructions

Movb

Move the value to respective memory address.

#### Mathematics

#### Number system

Possible Numbers that can be formed

For example, for binary number system, single byte can represent

For example, for decimal number system, three digits can represent

#### COMMUNICATION

Communication Mistakes

#### Uncategorized

Stack overflow mistakes

#### Programming

Productivity tools

Static Analyzer tool

Static Analyzer tool list for C++

#### GRAMMAR

### Tense

#### TENSE CHART

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Past | Present continuous | Preset prefect | Future tense |
|  | Had | have |  |  |
|  |  |  |  |  |

#### ETC

Using just

Sentence forming errors

Design

## COMMON PROGRAMMING MISTAKES

SEMICOLON AFTER A FOR LOOP:

STRAY IN the PROGRAM:

# DATA STRUCTURES

## TERMINOLOGY USED IN DATA STRUCTURE

Abstract data type(ADT):

What is difference between data structure and Abstract data type?

## LINKED LIST

### REVERSE THE LINKED LIST

### Rotate a doubly linked list

Problem

Examples

Algorithm

Pseudocode

## HASHING

### Hash functions

#### MESSAGE DIGEST

Properties

#### SHA (Secure Hash Al gorithm)

Properties

### WHIRLPOOL

Properties

### Miscellaneous

5381 is used for hashing the data.

### PROBLEMS

#### CONVERT LINKED LIST TO AS SAME AS ARRAY

Problem

Examples

Real world problems

Algorithm

Pseudocode

## GRAPH

Representation of graph:

Components involved in graph

Types of methods to represent graph:

Adjacency matrix:

Advantages:

Disadvantages

Adjacency List:

ADVANTAGES:

Disadvanges

## Tree

ADT

Data Structure Implementation:

How Tree Differs from graph:

|  |  |
| --- | --- |
| Tree | Graph |
| Tree is an special form of graph with some conditions. | Graph is not special form of tree |
| There is only one path between any two vertices in the tree. | There can be multiple path between any two vertices in the graph |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Tree Variants:

Planted Tree:

Classification of trees:

1. Based on how nodes are directed
2. Based on how nodes are ordered
3. Based on root of the tree

Ordered tree:

Fibonacci Tree:

Relation of Tree with Graph

Terminology:

Sub-Tree:

Number of SUB TREE OF A nodE:

Tree Elements:

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Tree based Data Structure:

Heap

Relation of heap with binary search tree:

Traversal in tree:

Classification based on the height and depth

Depth wise traversal

Inorder

Preorder

Postorder

Common uses of Tree

Common operations which can be performed in tree

Spanning tree

### REPRESENTATIONS

#### REPRESENTING TREE IN ARRAY

Formula

Note:

### AVL TREE

Why AVL Tree is better than normal binary search tree?

#### INSERTING NODE IN AVL TREE

Steps

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  | Remains in same position |
|  |  |  | Remains in same position |
|  |  |  |  |

#### SNAPSHOT OF INSERTION AND DELETION

|  |  |  |
| --- | --- | --- |
|  | INSERTION | DELETION |
| OPERATION | AVL ROTATION | AVL ROTATION |
| Z | Unbalanced node | Unbalanced node |
| Y | Child of Z in the travelled path | Greater height child of Z |
| X | Child of Y in the travelled path | Greater height child of Y |
| ANCESSTOR FIX | NOT REQUIRED | REQUIRED |

### RED BLACK TREE

Why RB tree over AVL tree?

Properties of Red black tree

#### INSERT A ELEMENT IN RED BLACK TREE

Steps:

#### DELETE IN RED BLACK TREE

Steps

#### SNAPSHOT FOR RED BLACK TREE INSERT AND DELETION

Let U be the node that is to be deleted and V be the node that replaced the node U.

|  |  |  |
| --- | --- | --- |
|  | INSERT | DELETE |
| OPERATION BASED ON | UNCLE of node U. | SIBLING of node U. |
| RED AND RED CASE | Mark both as black and mark grandparent of node U as red. | Do nothing |
| BLACK AND BLACK CASE | Do nothing | Check color of children of sibling.  One of the sibling children is red, perform rotation. |
| BLACK AND RED CASE | Do nothing | Check color of children of sibling.  Definitely both children of sibling should be black, perform double node rotation for node S and parent of Node S. |
| RED AND BLACK CASE | Perform rotation | Do nothing. |

## MISC DATA STRUCTURES

## BINARY SEARCH TREE

Delete a node:

Node with no children

Node with one children

Node with two children

## DATA STRUCTURES

### Terminology

Linear and non-linear data structure

Dynamic Data structure

Stack

When to use data structure?

Practical application of stack

### DIFFERENCE AMONG SIMILAR DATA STRUCTURE

#### Queue AND DEQUE DIFFERENCE

### BINARY TREE

#### Level order traversal

Algorithm

* 1. Go to step 2

#### B+ TREE

### REAL WORLD PROBLEMS

List of data structure to be used for real world programs

Graph

Backtracking

Tree

# ALGORITHM

## Analysis of alogorithM

### ASYMPTOTIC ANALYSIS

Difference between Big O and small o

## ALOGORITHM METHODOLOGY

### DYNAMIC PROGRAMMING

## REAL WORLD PROGRAMs

### SELECTION OF ALOGORITHM

Dynamic programming

## DATA STRUTURE ORIENTED

### FIND NEXT LARGEST ELEMENT IN ARRAY FOR ALL ELEMENTS

Problem

Examples

Algorithm

Pseudocode

### TREE BASED ALGORITHM

### STACK BASED ALGORITHMS

#### TERMINOLOGY

Infix Expression

Prefix Expression

Postfix Expression

#### INFIX TO POSTFIX

Steps:

#### INFIX TO PREFIX CONVERSIOn

Steps

#### REVERSE STACK USING RECURSION

Algorithm

#### SORT A STACK USING RECUSION

#### CHECK FOR BALANCED EXPRESSION

Algorithm

## SORTING ALGORITHM

### INSERTION SORT

Algorithm

### HEAP SORT

How heap sort works?

## TRAVELLING SALES MAN PROBLEM

## PROGRAMMING CONCEPTS BASED ALGORITHM

### STACK RELATED ALGORITHMS

### ARRAY RELATED ALGORITHMS

#### PRINT FROM KTH LARGEST NUMBER IN DECREASING ORDER

### TREE RELATED ALGORITHMS

#### CONVERT BINARY TREE TO LINKED LIST

### LINKED LIST RELATED ALGORITHMS

#### INSERT A NODE IN SORTED CIRCULAR LINKED LIST

#### REVERSE LINKED LIST IN GROUP

#### MERGE TWO SORTED LINKED LIST AS SORTED LINKED LIST

#### SPLIT CIRCULAR LINKED LIST IN TO TWO

# BATCH SCRIPTING

### PROPERTIES OF BATCH

File extenstion

### GRAMMAR

General commands

Echo

To turn off the echo statements

Dereferencing the variable

xcopy

Differences

Difference between % and %%

# PYTHON

Arithemetic operations in shell scripting

Loops in shell

Including header files for shell scripting

## DESIGN PRINCIPLES

## SOLID

## DESIGN PATTERNS

### DESIGN PATTERN TERMINOLOGY

Double Dispatch

### VISITOR PATTERN

How to avoid down casting in programs?

# EXCEL-VBA PROGRAMMING

## programming concepts

Comparision with other language

Sub-routine

Comment line

Variables:

Condtional Logic:

Range in VBA:

# MISC SUBJECTS

# BANKING

### TERMINOLOGY

LIQUIDITY

LIQUIDATION

Counter party

Risk weighted assets

Off balance sheet

Capital Adequacy ratio(CAR)

Equity

Financial Instruments

Financial security

Sub prime borrowers

### PAYMENT

#### Groups and corporations

NPCI – National payment corporation of India

NACH- National automated clearing house

Example:

AADAR PAYMENT BRIDGE:

IMPS – Immediate payment system

MMID- Mobile money identifier

UPI-Unified payment interface

Unique features

1. 24\*7 payment
2. Virtual address ([vinothcse123@axisbank.com](mailto:vinothcse123@axisbank.com)) for payment.

Bharat Bill payment system-BBPS

NATIONAL INSTITUTE OF BANK MANAGEMENT

#### CHEQUE

TERMINOLOGY

Legal amount:

Courtesy amount

CTS-Cheque truncation system

Types of cheque

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Ante date cheque:

Post date cheque:

Stale cheque:

Truncated cheque:

### CLASSIFCATIONS AND CATEGORY

#### CLASSIFICATION OF MSME ENTERPRISES

MSME – Micro small and medium enterprises

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

#### CLASSIFICATION OF FARMERS bASEd oN LAND

|  |  |
| --- | --- |
| FARMER | TYPE |
| Marginal farmer | <=1hectare |
| Small farmer | > 1hectare |

#### CLASSIFICATION OF BANKS BASED ON SIZE

Payment bank

Small finance banks

Properties

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

#### CLASSIFICATION OF ACCOUNTS IN BANK

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

### RESERVE BANK OF INDIA

Roles and Responsibility:

#### Types of accounts in bank

### HEADQUATERS

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |

### BANK ACCOUNT

#### TERMINOLOGY

INOPERATIVE ACCOUNT

### DEPOSIT ACCOUNTS

Senior citizen deposit

Demand liabilities

Time liabilities

Unclaimed deposits

### MONETARY & MONETARY RELATED

TERMINOLOGY

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

### TRADING

TERMINOLOGY

MONEY MARKET

Government instruments:

Interbank participation certificate(IBPC):

Treasury bills

CAPITAL MARKET

### LOANS & INTEREST

#### TERMINOLOGY

BASE RATE

MCLR(Marginal cost lending rate)

Exemption of MCLR:

DIR Loan(Differential rate loan)

#### TYPES OF INTEREST

REPO RATE

REVERSE REPO RATE

BANK RATE

Adjusted Net bank credit(ANBC)

Targets of PSL

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Phased manner achievement

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Limit of loans in PLS

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Outstanding target

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

RBI Revised PSL norms to RRB

Miscellaneous

#### MISCELLANEOUS

TAXATION ON INTEREST

Reverse mortgage loan

### TERMINOLOGY

Point of sale

EFTPOS- Electronic fund transfer at point of sale

SLA RATE-STATUATORY LIQUIDITY RATIO

CRR-CASH RESERVE RATIO

### MISCELLEANEOUS

#### BASEL NORMS

How Basel norms are formed?

BCBS

Major roles

Basel I Norms – Minimum capital requirements

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Basel II Norms

Three pillars

Basel III Norms – Disclosure and market discipline

Three pillars

Snapshot of Minimum capital requirement risk of all basel norms

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

#### SECTORS IN INDIAN ECONOMY

# OPERATING SYSTEM

## TERMINOLOGY

Kernel space & User space

Difference between kernel and operating system:

System call

### Kernel

## MATHEMATICS

### NUMERIC MATHEMATICS

Number types

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

### MISCELLANEOUS

#### TERMINOLOGY

ORDERED PAIR:

UNORDERED PAIR:

SET

Reason for pi constant:

#### SYMBOLS

|  |  |
| --- | --- |
| SYMBOL | DESC |
| \*\* | Power of |
|  |  |
|  |  |

## IPC – Inter process communication

### PIPE

### SHARED MEMORY

Disadvantages:

### MESSAGE QUEUE

Creating a message queue:

COMMANDS USED FOR MESSAGE QUEUE:

### REMOTE PROCEDURE CALL

### MISCELLANEOUS

Difference between message queue and pipe

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |

## MESSAGE QUEUE

MSGGET:

|  |  |
| --- | --- |
| PARAMETER | DESCRIPTION |
| key | It is the system wise unique identifier that was used to identify the queue.  If any body in the system wants to connect to same queue then then can use that identifier  Hardcoding a key will cause a problem because there is probability that same key will be used by anyone in the system. |
| msgflag |  |
| return value | Returns the message queue id on success and returns -1 on failures.  errno will be set upon failures. |

UTILITY FUNCTIONS:

ftok()

|  |  |
| --- | --- |
| PARAMETER | DESCRIPTION |
| char \*path | File path  This file should have the sufficient permission for the ipcs.  File should exists when calling this function, if not call will fail |
| char id | To create a bunch of Keys using the single file path |
| return value | return a newly generated key for the path and id specified.  The ftok will return the same key if path and id was passed as same.  It is unspecified that key will be same when path is deleted and created with same name.  On error it will return -1 and corresponding errno will be set. |

USEFULL COMMANDS:

# TRASH CAN

|  |  |
| --- | --- |
| Color | meaning |
| Green | Content is ready – Post to blogger |
| Yellow | Under development |
| OCEAN BLUE | Modified the content which is already posted to blogger |