**Task 5**

1. **What is CRM and ERP.**

| **S.No** | **ERP** | **CRM** |
| --- | --- | --- |
| 1. | ERP are the software solutions that helps organizations to manage their business processes. | CRM is the software that automates the customer communication with the organization. |
| 2. | It is a centralized system that streamline all the processes. | CRM is the single platform for turning customers into a potential client. |
| 3. | ERP is termed as the super set of SAP. | It is the subset of SAP |
| 4. | ERP is a web-based application. | CRM is a web based solution. |
| 5. | ERP systems are more focused about the organization growth and cost reduction. | CRM systems are focused about the customer relations with the organization. |
| 6. | They support the back office activities. | They support the front office activities. |
| 7. | Examples – NetSuite ERP, Scoro, AcTouch, etc. | Examples – Salesforce CRM, HubSpot CRM, Zoho CRM, etc. |

1. **What is SOTS.**   
   "Software on the shelf" is a term used to describe pre-packaged software that is readily

available for purchase and use without requiring any significant modifications or customization.

This type of software is typically designed to be applicable to a wide range of businesses or

organizations and may include features that are commonly used by many different types of

users.

The term "on the shelf" suggests that the software is physically available and ready to be used

immediately after purchase, much like a book or any other tangible product that can be

purchased off a shelf. This is in contrast to custom software, which is developed specifically for a particular organization or user and is tailored to their specific needs and requirements.

Examples of software on the shelf include popular productivity software like Microsoft Office or

Adobe Creative Suite, as well as customer relationship management (CRM) systems like

Salesforce or Hubspot.

1. **Cython code example.**

import run\_python   
import run\_cython  
import time  
  
n = 10  
  
start = time.time()  
run\_python.test(n)  
end = time.time()  
  
purePython\_time = end - start  
print ("Pure Python time = {}".format(purePython\_time))  
  
start = time.time()  
run\_cython.test(n)  
end = time.time()  
  
Cython\_time = end - start  
print("Cython time = {}".format(Cython\_time))  
  
print("Speedup = {}".format(purePython\_time/Cython\_time))

1. **Clean code vs Spaghetti code**

**Clean code** is a style of programming that emphasizes readability, simplicity, and maintainability. Clean code is easy to understand and modify, and it follows established conventions and best practices. Clean code is also well-structured, with logical dependencies and clear interfaces between modules.

**Spaghetti Code** is nothing but a generalized common-usage term for unstructured and difficult-to-read code. Such a type of code in any large code-base can create problems of its own, if not resolved on time. It can lead to a huge wastage of important resources like time and energy to find bugs and fix them because the code has no structure.

1. **What is paging and fragmentation.**

**Paging** is a technique used by operating systems to efficiently use memory resources by dividing memory into fixed-size blocks called pages. This allows the operating system to load only the pages that are needed, and free up memory when pages are no longer in use.

**Fragmentation**, on the other hand, refers to the problem of wasted memory space due to inefficient use of memory. There are two types of fragmentation: external and internal. External fragmentation occurs when free memory is broken up into small pieces that are too small to be used by any process, while internal fragmentation occurs when a process is allocated more memory than it needs, and the unused portion of the allocated memory is wasted.

1. **When the code can be slow.**

intentionally writing slower code is generally not a common practice, but there may be cases where slower code is preferred over faster code, depending on the specific requirements and constraints of the project.

1. Ex Testing and debugging: If the code needs to be thoroughly tested and debugged, it may be easier to do so with slower code that is easier to understand and modify than highly optimized code.
2. **What is the ram’s speed for virtual memory.**

The speed of RAM for virtual memory depends on various factors such as the type of RAM, the speed of the RAM, the size of the virtual memory, and the workload of the system. Virtual memory is a technique used by the operating system to use the hard disk as an extension of the RAM when the RAM is full. When the system runs out of physical memory, it uses the hard disk to store the data temporarily, which is slower than the RAM. Therefore, the speed of virtual memory is slower than the speed of RAM. However, the performance impact of virtual memory depends on the workload of the system and the size of the virtual memory. In general, it is recommended to have enough physical memory to avoid using virtual memory as much as possible to ensure optimal performance.

1. **How to inverse priority queue in python and c++.**

**Python**

import heapq

class InvertedPriorityQueue:

def \_\_init\_\_(self):

self.heap = []

def push(self, priority, item):

heapq.heappush(self.heap, (-priority, item))

def pop(self):

\_, item = heapq.heappop(self.heap)

return item

def \_\_len\_\_(self):

return len(self.heap)

**c++**

#include <queue>

template <typename T>

class InvertedPriorityQueue {

public:

void push(int priority, T item) {

queue.push(std::make\_pair(-priority, item));

}

T pop() {

T item = queue.top().second;

queue.pop();

return item;

}

int size() const {

return queue.size();

}

private:

std::priority\_queue<std::pair<int, T>> queue;

};

1. **How to create multi set and unordered set.**

from collections import Counter

# Create a multiset of integers

myMultiset = Counter([3, 1, 4, 1, 5, 9, 2, 6, 5, 3, 5])

# Create an unordered set of strings

mySet = set(['apple', 'banana', 'orange'])

1. **What is hash map table**.

A hash table, also known as a hash map or dictionary, is a data structure that maps keys to values. It’s an associative array that uses a hash function to compute an index into an array of buckets or slots, from which the desired value can be found. The hash function takes the key as input and returns an index into the array where the corresponding value is stored. Hash tables are commonly used for fast lookups, inserts, and deletes, with an average time complexity of O(1) for each operation. They are widely used in computer science and programming, including in databases, compilers, and operating systems.