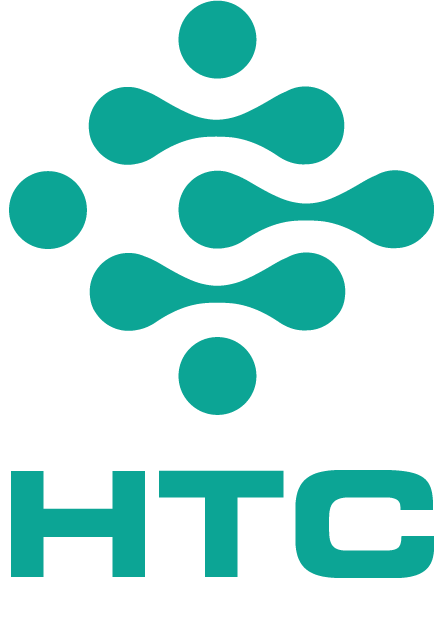
**JOB READINESS PROGRAM (JRP)**

**for Software Development & AI**

A Pikom x HTC collaboration

An Overview of the 12-week Program Syllabus



**Practical Assessment X**

**Table of Contents**

**Introduction 1**

**Week 1 - 4 2**

**Week 5 - 12 3**

**INTRODUCTION**

The purpose of this course serves to establish the following results:

* Illustrates the importance and interlink between Operating Systems, Data structures, Algorithms and Programming languages fundamentals.
* To think about efficiency in software development through the above interlink.
* To transform knowledge acquired in academics into a real-world environment.
* Designing applications that can manage vast amounts of data in a specialized area that may not be extensively covered by universities.
* Building applications with the awareness of testing from the start.  In all assignments, the instructor(s) reserve the right to intentionally cause the assignment to fail.

The reason for developing such a course is that it is observed that new graduates lack a connection between knowledge gained from university and how such knowledge is to be applied.

Building an application is not just about using relevant tools and coding.

| 1. For instance, a person who knows how to code can be given the best programming language and computing & communication tools, but still develop a program that will fail BIG time. Ie. A client requests some sorted data from a server and a bubble sort is used. |
| --- |

The above is an actual scenario during an interview session.  Some other observations of graduates:

| 1. Generating queue numbers in a hospital system with total disregard for concurrency control, resulting in duplicate queue numbers and chaos in the hospital. 2. Transferring files between systems with total disregard to the consumer/producer problem, resulting in database inconsistency in the data processing. |
| --- |

This link below illustrates further the point of why it is not just about coding and using the tools.

* <https://www.youtube.com/watch?v=rkZzg7Vowao>
* <https://www.youtube.com/watch?v=4RptzbNNoU0>

**Below is a brief description of what entails through the 12-week JRP syllabus:**

**Week 1: Programming Language**

This session attempts to provide guidance on how to learn a new programming language systematically.  It introduces the concepts of limits of the programming languages, such as storage size, internal representation, and the consequences of not understanding such limitations.  The performance of the language construct is evaluated. Manipulation of bitwise operators is reinforced.  Returning and accepting generic data types is discussed.

**Week 2: Database**

The database is not just about inserting, updating, deleting and inquiring about data.  The concept of a database transaction is discussed.  The consequences of not understanding this concept are deadly; namely: database inconsistency in a database recovery process.

Discussions will include table relationships, and how to avoid creating orphans. Effective use of various key(s) is discussed leading to producing sorted data without calling the sort function/verb.

Discussions also centered on the database data type in relation to the programming language data type and the limitations that go along with it.

This session serves to illustrate how knowledge of Data structures and Operating systems go hand in hand in developing applications.

**Week 3: Screen design**

Simple screen designs and their interaction with the database are introduced.  In conjunction with database and programming, the session will explore the behaviour of systems with hundreds or thousands of records/data.  It also serves to reinforce the mastery of the C# programming language.

**Week 4: A review of the various data structures and how they are implemented in a real-world** **environment.**  More advanced programming techniques will be in preparation for the actual implementation of such techniques in a real-world application, in conjunction with database knowledge, Operating systems and limitations.

This week starts to explore in detail concurrency and shared resources, and how an inadequate knowledge of these can lead to disastrous results in the application.

**Week 5:** **More discussions of concurrency controls** and shared data are discussed at the database level as to how applications should be built to avoid data inconsistency and database corruption.

Database recovery from a backup is discussed and practised, highlighting the importance of understanding the concept of a database transaction.

**Week 6:** **Theory exam week, followed by discussions on interprocess communications** and concurrency controls between processes, which is essential for distributed programming.

**Week 7:** **Getting data to and from a text file and database tables**.  What happens if the database structure does not correspond to the text data?

In this session, a discussion about testing is conducted, with the experience thus far of testing the candidate’s own application.  The concept of unit testing and integration testing is discussed as the merit of such testing methods.  Of interest is the percentage of ‘units’ in an overall application system.  Can a function that is supposed to be testing as ‘integration’ testing be done as a unit test?  Or it actually does not matter at all if a unit test or integration test is done?  How to think about testing and when does ‘testing’ start in the life cycle of application development?

**Week 8: Practical exam week, no lessons.**

**Week 9:** **Graph implementation in a multi-user environment.**  This culminates in applying knowledge of data structures and algorithms and operating systems concepts to developing real-world applications.

A look into why recursive algorithms will fail and techniques for converting a recursive algorithm into an iterative one.

**Week 10:** **Theory exam week followed by a discussion of next week’s exercise**: File transfer design.

**Week 11:** **File transfer design discussion**, which culminates in database knowledge, concurrency controls, and interprocess communications. The discussion also addresses anticipated system failures and how they are handled in the system, a further illustration of what testing is all about.

**Week 12: Practical exam week and summary/conclusion of the course.**

*Note: The syllabus provided is a tentative outline of the program which is supposed to cover the 12-week course. However, the topics covered and assignments given may vary depending on the performance of the students and may carry over from week to week.*