Software Development Approaches 1

# Materials Required

Moodle: <http://karabardec.mrooms.net/course/view.php?id=206>

Booklet: Preliminary Programming (on Moodle)

Textbook: Software Design and Development, the Preliminary Course (2nd Edition) by Samuel Davis

# Introduction

This lesson you will start working with the textbook. Contact your teacher immediately if you do not yet have a copy. You must also download and use the *Preliminary Programming – 3* booklet.

# Submission

Once you have completed the work for each week, save all of your work, which may include answers to questions as well as files and folders that contain the practical activities, in a folder named the same as the Weekly Worksheet number shown on your *Course Program*, for example WW1. ZIP the folder and upload the work for each week into the appropriate *Dropbin*.

# Tasks to Complete

1. **Read** pages 123 – 136 of the textbook and complete the following exercises and activities.
2. **Complete** Quiz 3A on Moodle. This is a self-marking quiz. Use the textbook to help you complete the answers. You have two attempts. In the second attempt it is expected that you would achieve 10/10.
3. **Read** *Repetition* and *Pre-test* *or guarded loop* pages 1 – 6 in the *Preliminary Programming – 3* booklet and complete Exercises 14, 15 and 16.
4. **Submit** this document in the **Software Development Approaches 1 Dropbin**

# Use Chapter 3A of the textbook to complete the following activities:

1. Historically, the structured approach was the recommended method for developing software. Describe reasons why this is no longer the case.

Because it’s slow and expensive

1. Describe what is done at each stage of the Software Development Cycle using the Structured Approach.

Defining and understanding the problem – System analysts are called in to create lists of inputs and then outputs, finally they create a development plan.

Planning and designing – The structures that will hold the data to be used during processing are designed. The project is then broken down into modules.

Implementing – The solution is coded in a programming language and the modules are handed to various programmers to be worked upon. Once completed the modules are merged to form the final solution.

Testing and evaluating – Tester teams check for any mistakes or glitches against the original requirements. It is handed out as an Alpha/Beta release to a few potential users to see how it holds up in a real life environment.

Maintenance – throughout the entire lifetime of the project (even after Beta testing) Maintenance teams will look after the software and system until it is discontinued.

1. Why is the quality of products developed using the prototyping approach likely to be lower than similar products using the structured approach?

Because the prototyping approach sacrifices quality for sheer quantity.

1. Compare and contrast the agile approach with the prototyping approach.

Prototyping approach allows for lots of early Beta testing thus making bug finding a lot easier compared to the agile approach that allows for very quickly made solutions that are highly adaptable to unexpected change requirements.

To contrast the two approach’s, you can compare them to a shapeshifter that can adapt to climates vs a duplicator that can make many copies of itself fast. In the end each have their uses and are highly effective when used in the right way.

1. The Structured Approach is often carried out by large teams of personnel with specific skills. Describe the tasks performed by system analysts, programmers and software testers.

Analysts create lists of outputs and inputs, Programmers work on creating modules of code and testers test the modules and try to find problems.

1. Complete Exercise 14 in *Preliminary Programming Booklet* *3*.

Use a WHILE loop to design an algorithm which reads the names of students in a class and keeps a count of the number of students. When the user inputs ZZZ, the number of students in the class is displayed.

Don’t understand how.

1. Complete Exercise 15 in Preliminary Programming Booklet 3.

Use a WHILE-DO loop to write a PASCAL program named *Authentication* which asks users to input their age. If they are under 18 they are told they are too young and the program asks again for the user to input their age. If they are over 18 they are told that they are able to enter.

PROGRAM Authentication1(output, input);

VAR

UserAge: integer;

BEGIN

WRITELN('What is your age?');

READLN(UserAge);

WHILE UserAge < 18 DO

BEGIN

writeln('Only ',UserAge,', you are too young!');

REPEAT

WRITELN('What is your age?');

READLN(UserAge);

UNTIL UserAge > 18

END;

BEGIN

WRITELN('Come into my parlour');

END;

READLN;

END.

1. Complete Exercise 16 in *Preliminary Programming Booklet 3*.

Save and run *PreTest*. Place all files and folders into the WW folder along with your work for this week.

# Assessment Task 1

* Use *Gantt Project* to create a Gantt chart that identifies the tasks you will need to complete and estimates when they will be completed from now until when AT1 is due. Use the *Project Guidelines* document on Moodle to help you complete this task. Make sure you go through the *Task Description* and *Task Instructions* as well as the *Marking Guidelines* and include all of the things that that the task requires you to do.
* Add what you have done this week to your logbook.

s