Sample projects

Below are three ideas for programming projects that you could develop incrementally, building more features into them as you progress through the book, learning new concepts.

Suggestion 1: Multiple choice Quiz Program

Chapters 1-4
- Includes Screen output, Keyboard input.

Example: Prints a series of multi-choice questions to the screen asking the user to supply answers.

Chapters 1-5
- Includes Decisions and calculation, and all constructions above.
- Comments included
- Indentation.

Example: Prints a series of multi-choice questions to the screen asking the user to supply answers. Indicates whether the answers were correct.

Chapters 1-6
- Includes Loops, and all constructions above.

Example: As above, and in addition prints out a mark at the end. Uses a loop to allow the user to play repeatedly (with the same questions).

Chapters 1-7
- Includes Loops within loops and all constructions above.

Example: As above, except the program gives the user a chance to try again immediately if they give a wrong answer. Prints out the total number of attempts at the end.

Chapters 1-9
- Includes Arrays, and all constructions above
- Program split into methods with well-defined tasks
- Methods individually commented.

Example: As above, but stores the answers given in an array with one entry per question.

Chapters 1-10
- Includes File input or output, and all constructions above
- Program split into separate methods with information passed via arguments.

Example: As above, but outputs the answers given with the total score to a file.

Other ideas: The questions and answers are read in from a file. The user is given a choice of a series of question topics - each set of questions and answers is stored in different files and the one requested is loaded in.

Suggestion 2: Noughts and Crosses

Chapters 1-4

- Includes Screen output, Keyboard input.

Example: Inputs 9 characters storing each in a different variable and prints out a corresponding noughts and crosses board (use a dot for empty). For example, if the following series of characters were input:

```
. . . . O . . X .
```

the following should be printed:

```
. . .
. O .
. X .
```

Chapters 1-5

- Includes Decisions and calculation, and all constructions above.
- Comments included
- Indentation.

Example: Asks user for a position and prints a new noughts and crosses board with an X in that position. For example, if the user input 7, and the above board had been input, the following new board should be printed.

```
. . .
. O .
X X .
```

Chapters 1-6

- Includes Loops, and all constructions above.

Example: Allows two people to play a single game of noughts and crosses using the computer as the board. No correction of mistakes is made, or detection of a winning position reached. It stops when an invalid position is input. Uses 9 variables to represent the board – each recording the piece in a different board position.

Chapters 1-7

- Includes Loops within loops and all constructions above.

Example: Allows two people to play a single game of noughts and crosses using the computer as the board. Detects an invalid position and repeatedly asks for a new position in that case.

Chapters 1-9

- Includes Arrays, and all constructions above
- Program split into methods with well-defined tasks
- Methods individually commented

Example: As above, except the board is now recorded in a single array variable.

Chapters 1-10

- Includes File input or output, and all constructions above
- Program split into separate methods with information passed via arguments.

Example: As above, but in addition a partially completed game can be saved to a file to be resumed later.

Other ideas:

Have the program play by picking random positions to go in. Alternatively, make the program play from a pre-programmed strategy (e.g. go for centre first, look for winning positions to block, etc. OR some positions are worth more than others - find a position to maximise the

score). The program keeps a record of previous positions encountered which led to a defeat in the next go and does not make the same move again – i.e. it learns from its mistakes!

Suggestion 3: Examiner's Dilemma Game

Write a program to allow two people to play the game of *Examiner's Dilemma*. A variation on this game has been used to do research into co-operative behaviour by writing programs to try different strategies. *A student is taking monthly exams and offers to bribe an examiner to give them each paper in advance. The examiner is to leave the paper under a bench in the park. The student will leave the money under a second bench at the same time. They then each go to the other bench to get the money/paper respectively. On each round of the game one of 4 things may happen: 1) The paper and money are both left (both score 5 points) 2) The paper is left but the student cheats this time and leaves no money (examiner 25 points, student 0) 3) The money is left but the examiner cheats and leaves no paper (examiner 0, student 25) 4) Neither money nor paper is left (both score 15 points). The aim is to get the lowest score over 5 rounds.*

Chapters 1-4

- Includes Screen output, Keyboard input.

Example: The program asks the examiner if they will leave the paper (1 = YES, 0= NO) and prints out a statement saying what they have done.

Chapters 1-5

- Includes Decisions and calculation, and all constructions above
- Comments included
- Indentation.

Example: The program asks both examiner and student for their choice, and prints out a message saying what each person chose to do and then prints out the scores of each. Clear the screen between each person's turn so that the second person doesn't see what the first person typed.

Chapters 1-6

- Includes Loops, and all constructions above.

Example: Allows two people to play a series of 5 rounds of examiner's dilemma using the computer as the referee and to keep each person's score.

Chapters 1-7

- Includes Loops within loops and all constructions above.

Example: Allows two people to play a series of rounds using the computer as the referee. Detects an invalid choice (a number other than 1 or 0) and repeatedly asks for a new choice in that case.

Chapters 1-9

- Includes Arrays, and all constructions above
- Program split into methods with well-defined tasks
- Methods individually commented.

Example: As above, except the score of each round is recorded in two array variables (one for the student and one for the examiner). At the end the score for each round is printed out.

Chapters 1-10

- Includes File input or output, and all constructions above
- Program split into separate methods with information passed via arguments.

Example: As above, but the results of each round can also be saved to a file to be resumed later.

Other ideas: Have an option for playing the computer, where the computer co-operates then does the same as the opponent did last round.