Centre Number						Candidate Number				
Surname										
Other Names										
Candidate Signature										

GCSE
For submission in 2015

Computer Science 4512 4512/CB3

Unit 4512/1 – Practical Programming

Scenario 3 Game Application

AQADo

For candidates entering for the 2015 examination
To be issued to candidates on or after 5 June 2013

This scenario is one of four available. Each of the four scenarios is available in a separate candidate booklet. You must complete ***two*** *of the four scenarios.*

- You have approximately 25 hours in which to complete this scenario.
- Before starting work on the problem, read the whole of this Candidate Booklet thoroughly. You can ask your teacher to explain anything in this booklet, except Computer Science specific terms, that you do not understand.
- There are restrictions on when and where you can work on this problem. Your teacher will explain them to you. For example, you should only do work that you intend to hand in for marking when a teacher is present, so that he or she can confirm that the work is your own. The Candidate Booklet must **not** be taken outside your school/college.
- You may need to use the Internet to research certain parts of the problem. This does not have to be within the 25 hours recommended time.
- You will need to complete and sign a Candidate Record Form which your teacher will provide.

Information

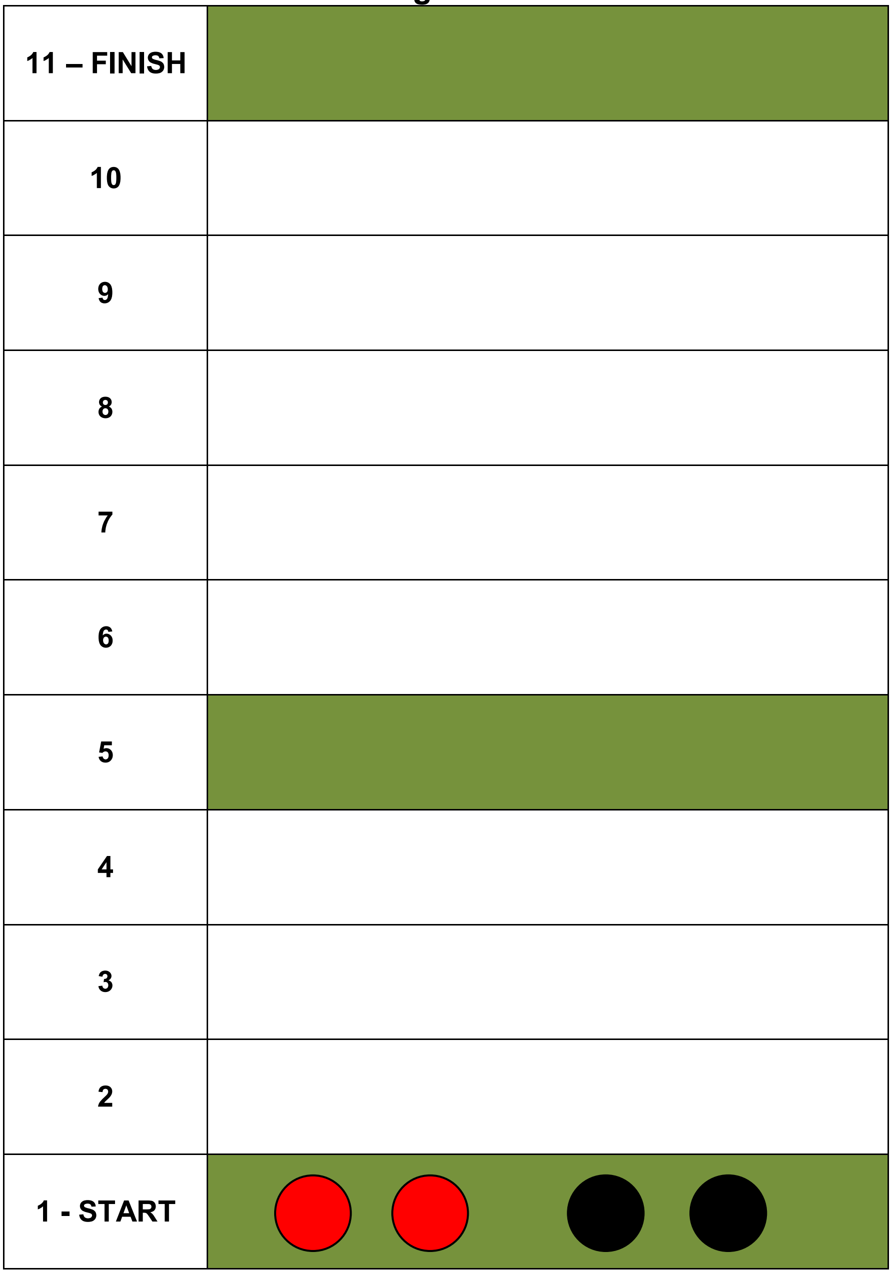
You will also be marked on your use of English. It is important to:

- make sure that all your work is legible
- use correct spelling, punctuation and grammar
- use a style of writing which suits the person you are writing for
- organise your information clearly, so that you make yourself understood
- use Computer Science terms where they are needed.

Scenario 3: AQADo

AQADo is a turn-based game where two players compete to move both of their pieces to the end of the board. The board consists of 11 spaces. Each player has two pieces. At the start of the game all the pieces are placed on the first space of the board (START). The first, fifth and eleventh spaces on the board are safe spaces. **Figure 1** shows the state at the start of a game. In **Figure 1** the players' pieces are shown as black or red circles, safe spaces are highlighted in green and the board is shown with the starting position (START) as the bottom space and the finishing position (FINISH) as the top space.

Figure 1



The players take it in turns to roll a die. A four-sided die is used when playing AQADo. The result of the die roll determines the move that a player is allowed to make on the board.

Table 1 shows the allowed move associated with each possible result of the die roll.

Table 1

Die number	Action
1	Move one of your pieces one space nearer to FINISH
2	Move one of your pieces two spaces nearer to FINISH
3	Move one of your pieces three spaces nearer to FINISH
4	Move one of your pieces one space back towards START

If, as a result of a move, a player's piece lands on a space containing an opponent's piece then the opponent's piece is moved back to START **unless** the space landed on is a safe space; a piece on a safe space cannot be sent back to START.

A player cannot move one of their pieces onto a space already occupied by their other piece unless the space is a safe space.

A piece cannot move backwards if it is on START or forwards if it is on FINISH.

If a player can make a move then they must do so (eg if they have one piece on START and one piece on FINISH and they roll a four on the die they must move the piece on FINISH back one space).

If a die roll allows a player to move either of their pieces they may choose which **one** of their pieces they want to move.

If a move would take a piece past FINISH then the piece is moved to FINISH (eg a piece on space nine which is moved three spaces nearer to FINISH would land on FINISH).

If a player cannot make a move then their turn is over and the other player has their turn.
A player wins the game when both of their pieces are on FINISH.

Turn over for the tasks

Tasks

1. Develop a main menu for the program. The options on the main menu should be:
   - Enter player names
   - Play game
   - Quit

2. Develop the part of the program where the players enter their names. After both players have entered their names the program should return to the main menu.

3. Develop the part of the program so that when the 'Play game' option is selected from the main menu the initial state of the game is displayed (as shown in **Figure 1**). The board and pieces do not have to be shown in exactly the same way as has been used in **Figure 1**. Any display that clearly shows the 11 spaces on the board, the current positions of the four pieces and which pieces belong to which player is fine.

4. Develop the part of the program that generates random numbers to simulate the rolling of a four-sided die. The program should tell the player the result of the die roll and tell them the action (as described in **Table 1**) that corresponds to the result of the die roll. The program should use the player's name when displaying the message. If the players have not entered their names then the default names 'Player 1' and 'Player 2' should be used.

5. Develop the part of the program that checks if the result of the die roll will allow a player to move either of their pieces this turn. If no move is possible then an appropriate message should be displayed and the player's turn is finished.

6. Develop the part of the program that allows the player to select which one of their pieces they want to move. If the result of the die roll means that the piece selected cannot be moved then an appropriate message should be displayed and the player should be allowed to select their other piece. A piece cannot be moved if:
   - the space the piece would be moved to is already occupied by the same player's other piece (unless it is a safe space)
   - the piece is on START and the result of the die roll is a four
   - the piece selected is one of the opponent's pieces
   - the piece is on FINISH and the result of the die roll is not a four.

7. Develop the part of the program that makes the legal move selected by the player. If the piece lands on the same space as an opponent's piece(s) then:
   - if it is a safe space then no other pieces are moved
   - if it is not a safe space then the opponent's piece should be sent back to START.

   The program should then display the board showing the new positions of the pieces.

8. Develop the part of the program that checks if a player has won the game after each move. A player has won the game if both of their pieces are on FINISH.
   - When a player wins the game an appropriate message that includes the player's name should be displayed. The program should then return to the main menu.
   - If the game has not been won then it becomes the other player's turn. Tasks 4 to 8 should then be repeated until one of the players has won the game.

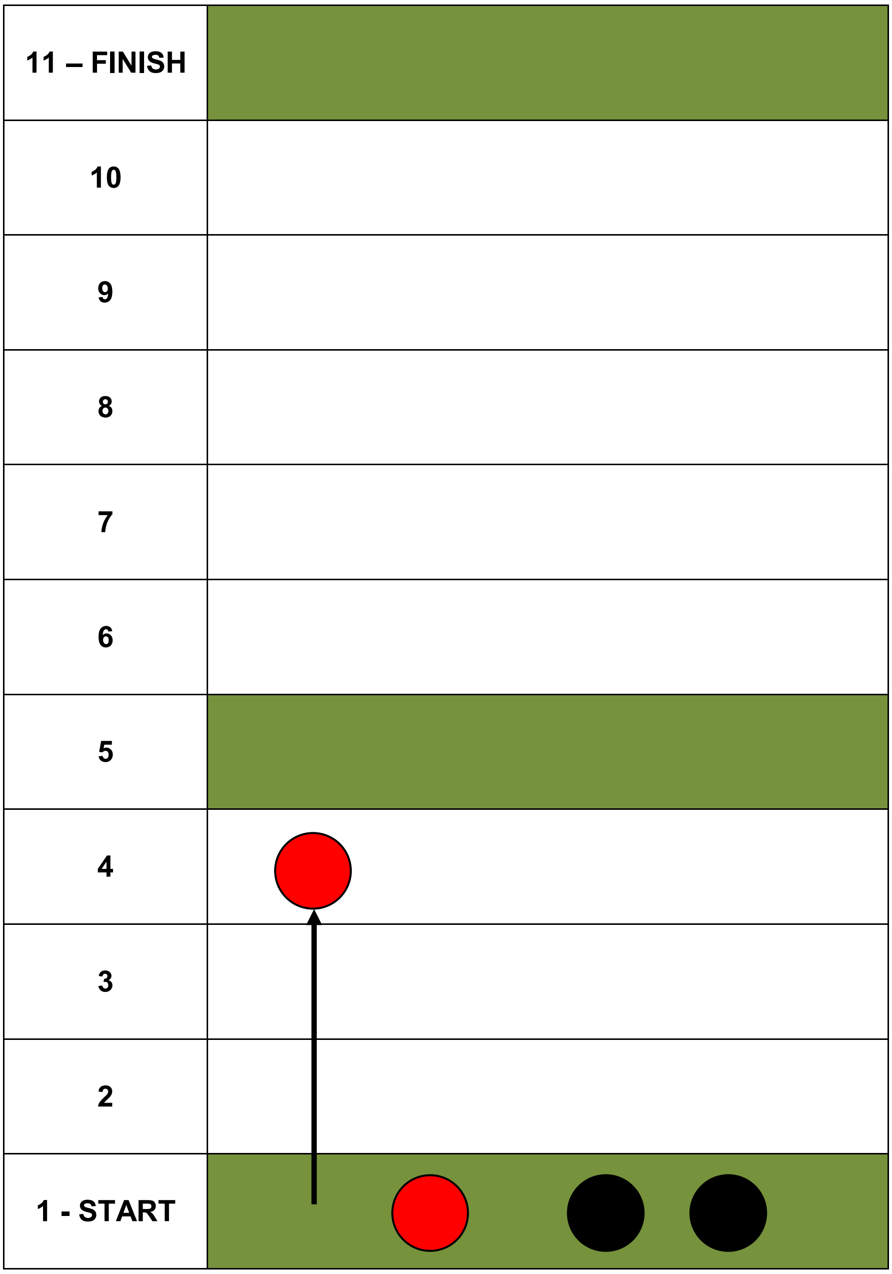
9. Extend the program so that it uses a five sided die. If a player rolls a five then they can select one of their pieces and move that piece to the next unoccupied space after the space it is currently on. If there are no unoccupied spaces after the space the piece is currently on, then that piece cannot be moved.

Turn over for an example of the game in action

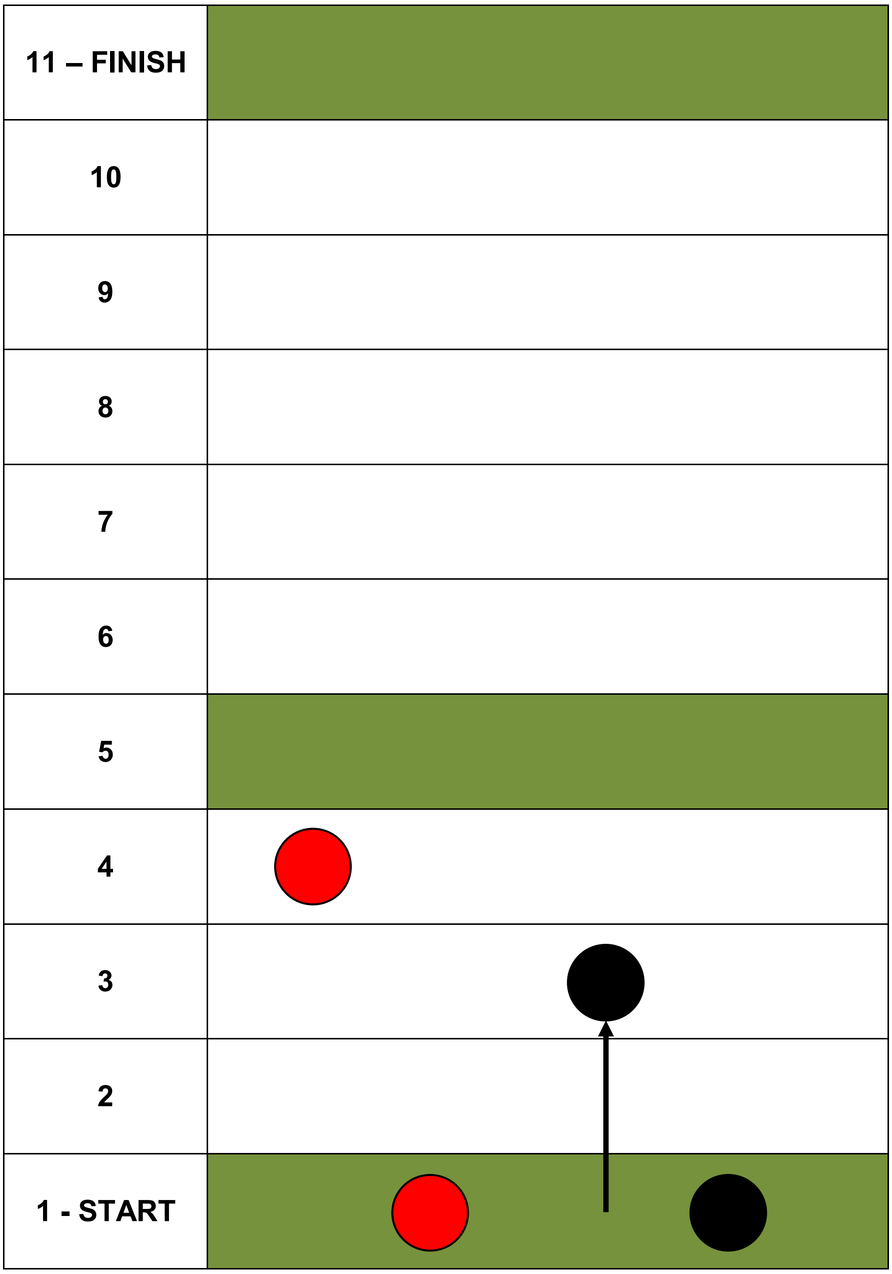
An example of the game in action

Player 1 has the red pieces and goes first. Player 2 has the black pieces.

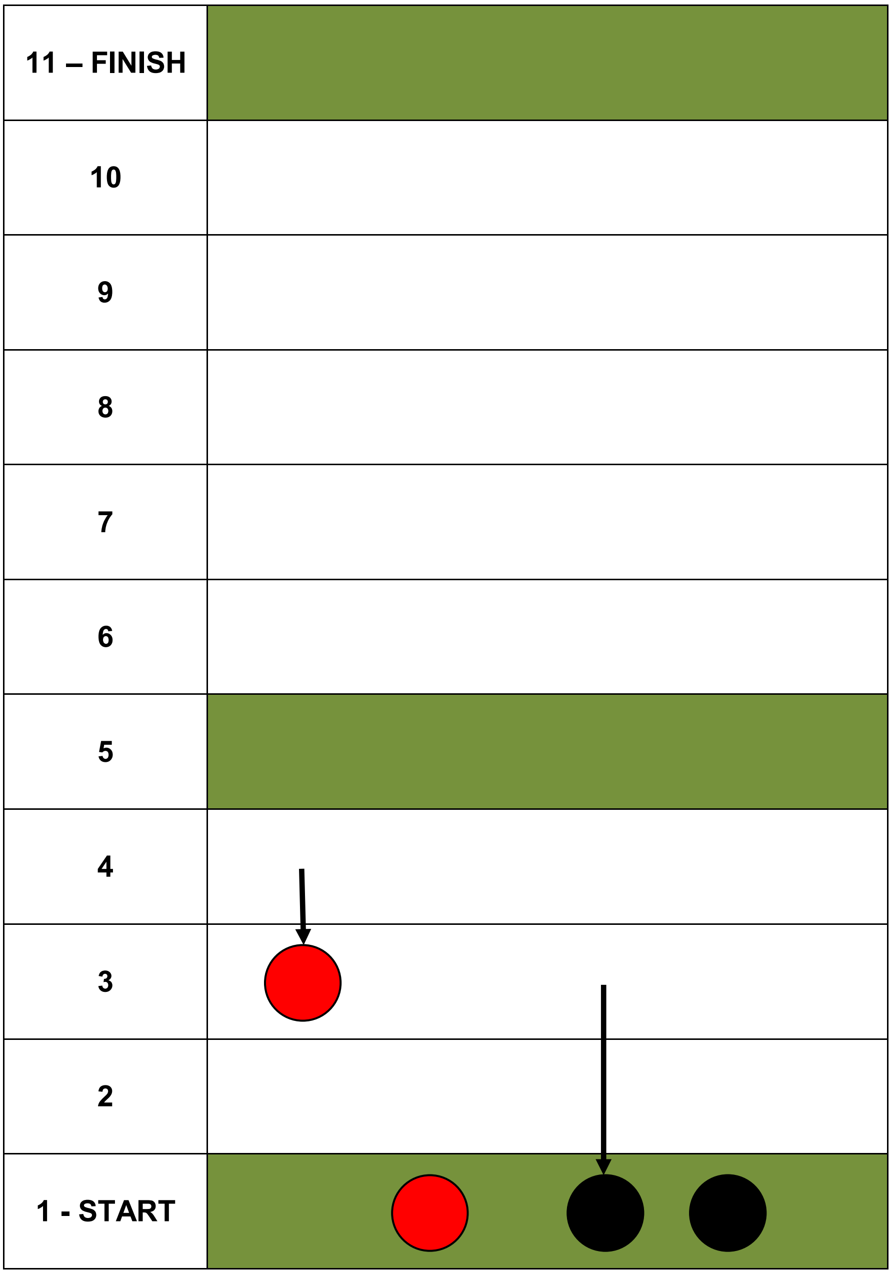
Player 1 rolls a three – the player selects one of their pieces and moves it three spaces nearer to FINISH.



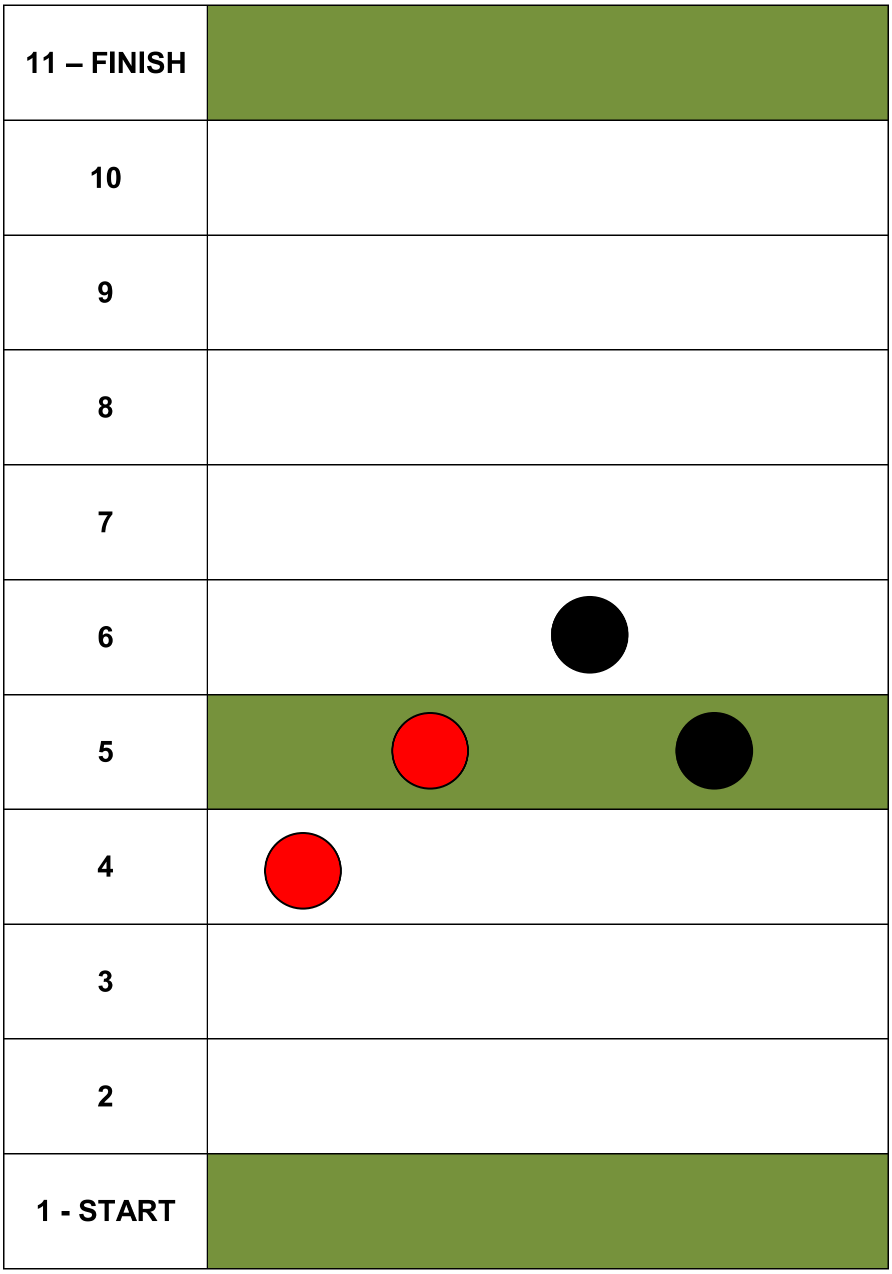
Player 2 rolls a two – the player selects one of their pieces and moves it two spaces nearer to FINISH.



Player 1 rolls a four – they can only move one of their pieces, as their other piece is on START and cannot move backwards. Player 2’s piece is then sent back to START.



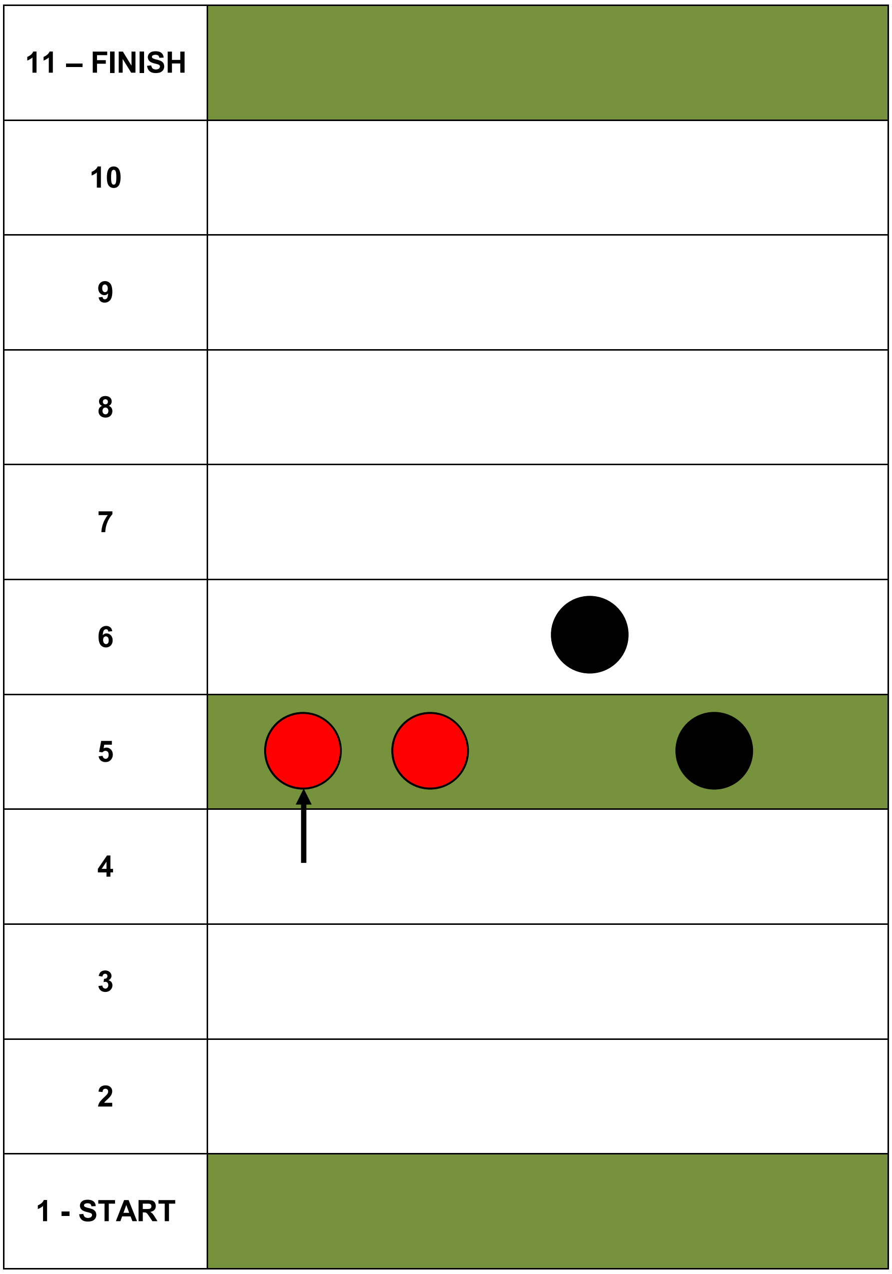
A few moves later the board position is as shown below



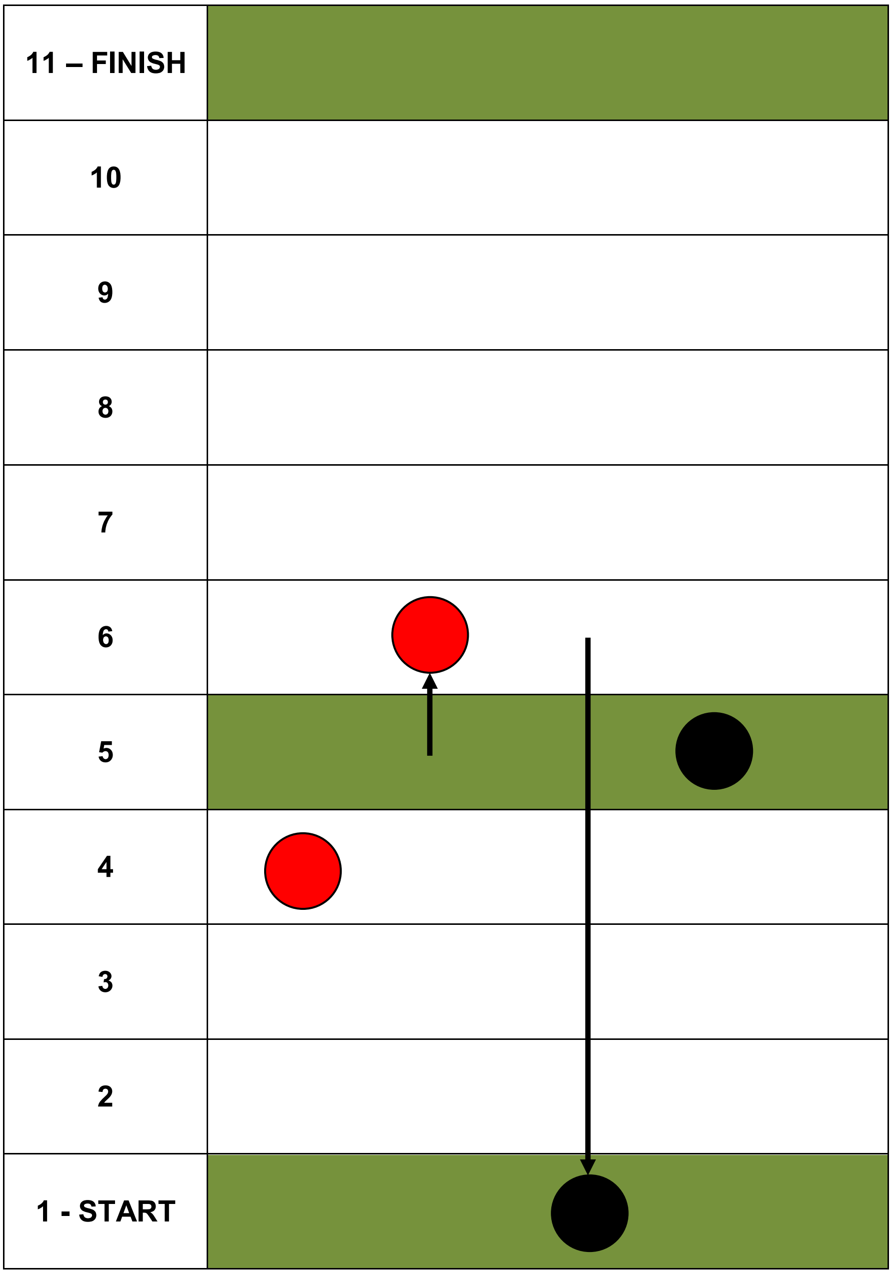
Player 1 now rolls a one and has a choice of moving:

1) one of their pieces to the safe space at Space 5
2) the other piece to Space 6 which will send the opponent's piece back to START.

Option 1 – move piece to Space 5



Option 2 – move piece to Space 6

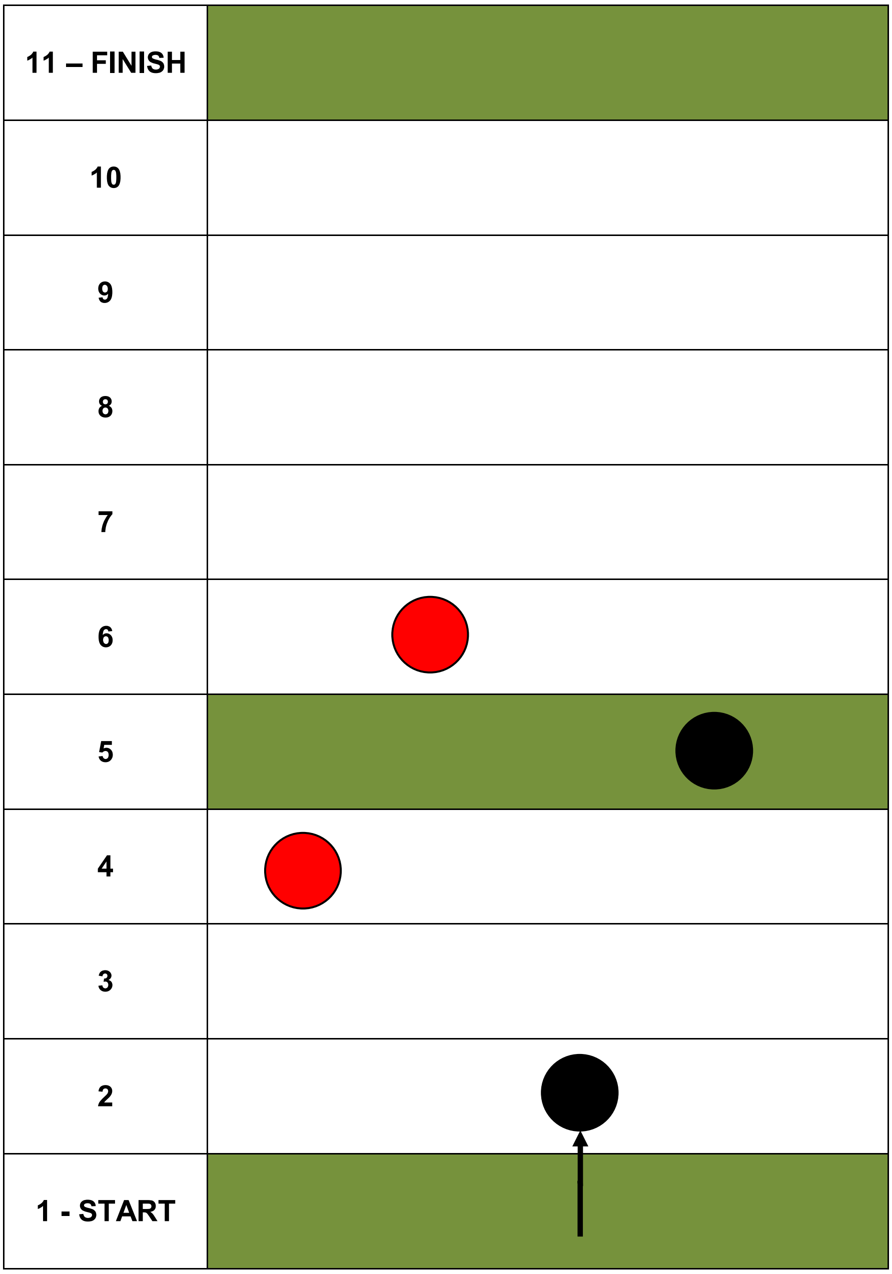


An example for Task 9

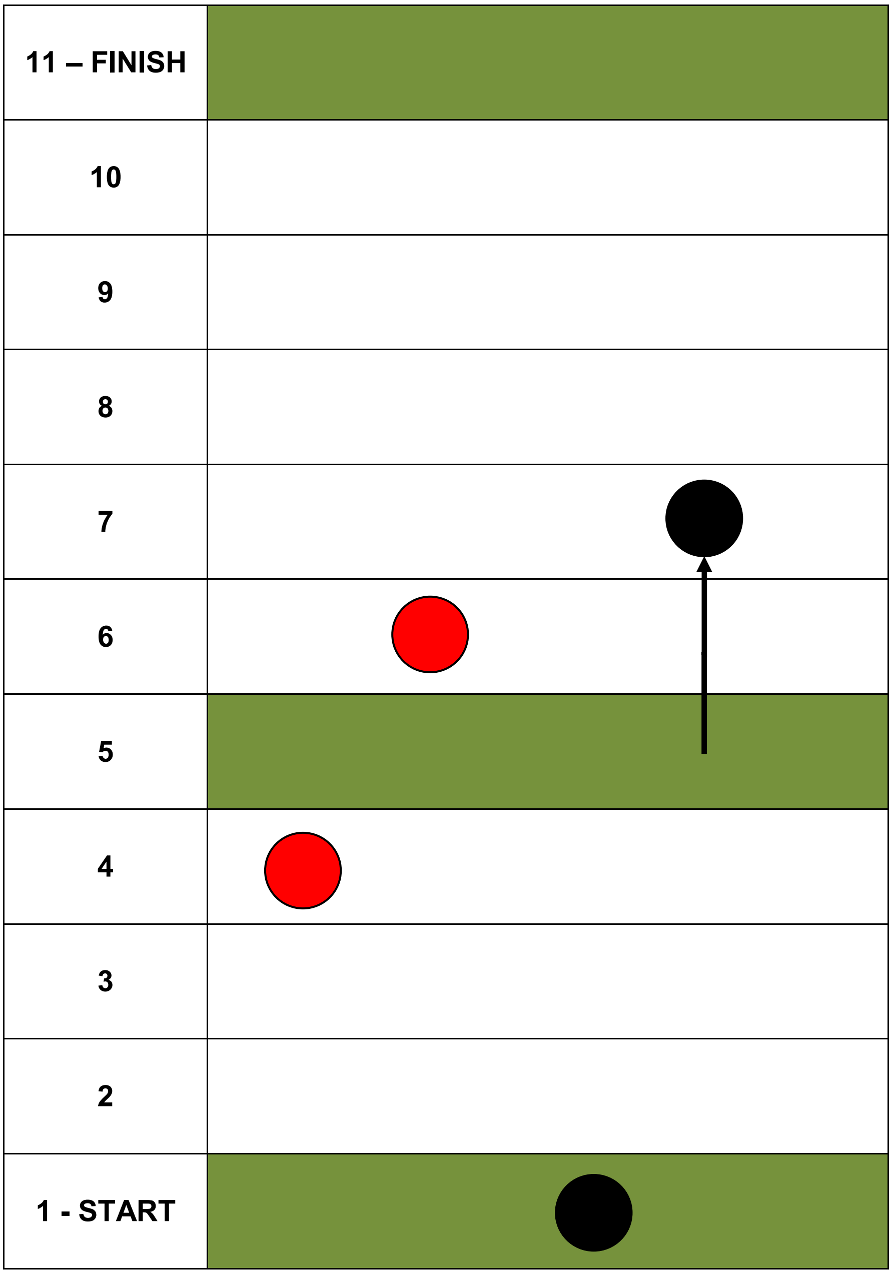
Player 2 now rolls a five and must move one of their pieces to the next unoccupied space. They have a choice of moving:

1) one of their pieces to Space 2
2) the other piece to Space 7.

Option 1 – move piece to Space 2



Option 2 – move piece to Space 7



Turn over for information on organising your portfolio

Turn over ►

In addition

1. Your Portfolio

Remember that you are looking to provide an application that will allow a person to fully play the game.

You are free to use whatever software tools and techniques are available to you.

What your teacher will be looking for and how to provide that evidence for your Portfolio

In preparing you for this unit of work, your teacher will have provided you with more information about the section headings below.

Part 1 – Design of solution

Design of solution (0–9 marks available)
What you must do
• Show an understanding of what the problem involves with reference to the user's needs. • Produce an overview plan that shows how the problem is to be solved. • Produce pseudo code (or suitable alternative) showing the main blocks within the proposed solution.

Part 2 – Solution development

Solution development (0–9 marks available)
What you must do
• Show evidence of an understanding of how the final solution meets the needs of the user. • Produce annotated code that demonstrates an understanding of the programming techniques used.

Part 3 – Programming techniques used

Programming techniques used (0–36 marks)
What you must do
• Show an understanding of the programming techniques used and how the different parts of the solution work together. • Explain/justify the choice of programming techniques used to create a solution that has been coded efficiently. • Show evidence for the purpose and use of data structures. • Show the techniques used (appropriate to the language used) within the code to make the solution robust.

Part 4 – Testing and evaluation

Testing and evaluation (0–9 marks available)
What you must do
• Produce a test plan that shows the expected tests, test data and expected results. • Show that the planned tests have been carried out and provide a record of the actions taken. • Evaluate how the final solution meets the needs of the user.

2. Organising your Portfolio of work

Your Portfolio is where you keep the evidence that you have produced.

You should imagine that the Portfolio is to be used by another person who is interested in how you produced your solution. It is to help them to do something similar. It is important that you organise work for the Portfolio as shown below.

- You must keep all the work you produce for the organiser in hard copy in a Portfolio (or save your work electronically in folders which you will later copy onto a CD or DVD). Your teacher will have instructed you what to do.

- If you are putting hard copy printouts in your Portfolio make sure that you number each page and fasten it all together. Take your work out of any plastic sleeves before you hand it in to your teacher for marking.

- Each page should have your name, centre number and candidate number clearly shown on it.

- When you have completed this scenario, if you are putting your work on a CD or DVD, put the work for each heading on page **14** of this booklet in a separate folder. Each folder must be clearly named *(for example, 'Design of solution', 'Solution development' etc)*. Inside each folder the work must have a filename (*for example, 'What the problem involves', 'Control statements' etc*) which should be a final version for each heading. The CD or DVD should have your name, centre number and candidate number clearly shown on it. Your teacher will have advised you what to do.

END OF CANDIDATE BOOKLET