Name: Class:

Task 1

Edward has drawn a flowchart to show the basic steps in a program for a game. A player continues to roll two die, adding the numbers rolled to his score. The player can stop at any time, but if he rolls a double before stopping his score goes back to zero.

(a) Use the flowchart to write a program in the language of your choice, from this algorithm.

score = 0

score = score + die1
+ die2

die2 = random(1,6)

Another go?

Output score

Start

End

die1==die2?

Yes

Yes

No

die1 = random(1,6)

Score = 0

No

(b) Write down up to three examples of syntax errors that you made in your program as you developed it.

(b) Correct these errors and make sure your program works correctly.

(c) Insert three deliberate syntax errors in your program. Write them down below.

(d) Now insert a logic error in the program. Test it yourself to see the effect. Write the error and the expected result below. Swap with a partner and ask them to find and correct the error.

(e) Swap with a partner and see if you can each find the other’s syntax errors without compiling the program. Write down the syntax errors that you found.

(f) Write down the logic error that your partner added to their program.

Task 2

The pseudocode program shown below is intended to model the calculation of the toll charge for crossing a bridge on a motorway. Between 6am (0600hrs) and 10pm (2200 hrs) inclusive, the charges are as follows:

* Motorcycle £1.00 (Vehicle type 1)
* Car £2.00 (Vehicle type 2)
* Goods vehicle £2.50 (Vehicle type 3)
* HGV and coaches £5.00 (Vehicle type 4)

After 10pm and up to but not including 6am, there is no charge.

Amanda writes the following pseudocode:

 if time <= 0600 OR time >= 2200 then

 charge = 0

 else

 switch vehicle:

case 1:
charge = 1.00

case 2:
charge = 2.00

case 3:
charge = 2.50

case 4:
charge = 5.00

 default:

 charge = 0.00

 endswitch

 endif

 print(charge)

Complete the following test plan.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test number** | **Data (Vehicle type)** | **Data (Time)** | **Reason for test** | **Expected result for Charge** | **Actual result** |
| 1 | 1 | 1245 | Valid time | 1.00 | 1.00 |
| 2 | 2 | 0600 |  |  |  |
| 3 |  |  |  |  |  |
| 4 |  |  |  |  |  |
| 5 |  |  |  |  |  |

(a) Are your tests sufficient to thoroughly test the program? What assumptions have
been made?

(b) Is the logic of the program correct? If not, what should be changed?

Task 3

The following algorithm performs a binary search to find an item in a sorted array A of length n. If the item is found, its position in the array is displayed. If the item is not in the list, “Item not found” is displayed. Assume that the first element of the array is A[0].

 itemFound = False

 searchFailed = False
Array names[13]
names = ["Anna", "Bill", "David", "Faisal", "Jasmine", "Jumal", "Ken",
 "Michela", "Pavel", "Rosa", "Stepan", "Tom", "Zac"]
 first = 0

 last = names.length – 1

 do

 midpoint = int((first + last)/2)

 if names[midpoint] == itemSought then

 itemFound = True

 else

 if first > last then

 searchFailed = True

 else

 if names[midpoint] < itemSought then

 first = midpoint + 1

 else

 last = midpoint - 1

 endif

 endif

 endif

 until itemFound OR searchFailed

 if itemFound then

 print("Item is at position " + str(midpoint))

 else

 print("Item is not in the array")

 endif

(a) The user searches for the name Tom. Complete the trace table below

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **itemFound** | **searchFailed** | **first** | **last** | **midpoint** | **names[midpoint]** | **OUTPUT** |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

(b) The user searches for the name Erik. Complete the trace table below

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **itemFound** | **searchFailed** | **first** | **last** | **midpoint** | **A[midpoint]** | **OUTPUT** |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

Task 4

The following pseudocode is supposed to sort an array of names into ascending alphabetical order.

1 userName = ["Carl","Tamsin","Eric","Zoe","Alan","Mark"]

2 numItems = 6

3 while numItems > 1

4 for count = 0 to 4

5 if userName [count] > userName[count+1] then

6 temp = userName[count+1]

7 userName[count] = userName[count+1]

8 userName[count+1] = temp

9 endif

10 next count

11 numItems = numItems – 1

12 endwhile

13 print (userName)

However, when the program is run, it produces the following output.

'Alan', 'Alan', 'Alan', 'Alan', 'Alan', 'Mark'

(a) What sort of error is this?

 (b) Describe how you might set about finding the error in the program.

(c) Which line is incorrect? What should it be?