Computer Science – Drills and Skills

The questions that follow are designed to test and build up pseudocode and/or programming skills with students. They can be used as worksheets for students to work through at their own pace, or can be used by the teacher to lead a lesson using mini-whiteboards. Various approaches are discussed in the breakout session and the magazine from the PiXL Computer Science conference 2019. As with a questions, please adapt them to suit the needs of your students and your style of teaching as appropriate.

Part of the point of these questions is to build fluency and mastery through repetition of the fundamental programming skills of sequencing, selection and iteration. In some case this is done directly, as there is repetition built in to each set of questions. In other cases the repetition is embedded in another exercise, for example the later exercises on array include elements of both selection and iteration. If I am using mini-whiteboards, I will do this as I go, adding or skipping questions depending on the response from the group.

Once you get the idea, I hope that you will feel enabled to write your own sets of questions. For example, you can copy and paste a set of questions from this resource and simply change the conditions to get a new set of practice questions on the same topics. You can also combine the exercises given here to get a new exercise. For example, the 2D array exercise currently uses test scores and there is a separate exercise using item prices to find a total price for a customer. These could be combined to give a new 2D array exercise where the goal is to (eventually) find the total price for each customer.

You may wish to apply these principles to new topics or questions if students get stuck. For example, if you have a challenging exam question, can you break it down into distinct skills, such as a specific loop and a specific if statement? If so, can you write 5 questions on each skill, building up to combining the skills in the exam question?

For students who have already mastered these skills, the following extension questions can be applied to most of the practice exercises:

1. Identify all the programming constructs used in this sequence of questions
2. What question(s) would you ask using the variable(s) provided? Write a model solution for your question
3. How many different ways can you solve the last problem in the sequence? Which is more efficient and why?
4. Write a mark scheme for the last problem in the sequence
5. Design a test plan for the last problem in the sequence
6. Can you turn your code fragment into a full program with correct inputs?
7. Can you re-write your code fragment as a function?
8. Can you re-write the questions using your own scenario?

Students should be encouraged to not only solve each problem, but to do it accurately and quickly, and I recommend using the pseudocode standard for your exam board.

Finally, please experiment with the questions. The goal is to develop fluency, so whatever the student needs to do, they have the confidence to try. So don't just go through a list in order, go through forward, backwards, every other item, backwards in threes, and so on until the student has complete freedom to code whatever they need.

Selection Statements

Set 1
Assume that user has already been asked for their score on a test, out of 100. This has been validated and stored in a variable called *score*.

Write a code fragment to

1. print “yes” if the score is greater than 50
2. print “yes” of the score is greater than 90
3. print “yes” if the score is greater than 50 and “no” otherwise
4. print “pass” if the score is greater than 50 and “fail” otherwise
5. print “merit” if the score is greater than 75, “pass” if the score is greater than 50 and “fail” otherwise
6. print “distinction” if the score is greater than 90, “merit” if the score is greater than 75, “pass” if the score is greater than 50 and “fail” otherwise

Set 2
Assume that user has already been asked to input a string. This has been validated and stored in a variable called *word*.

Write a code fragment to

1. Print the length of the word
2. Print “long enough” if the word has 8 or more characters
3. Print “long enough” if the word has 8 or more characters and “not long enough” otherwise
4. Print “good” if the word has 8 or more characters and “weak” otherwise
5. Print “strong” if the word has 12 or more characters, “good” if the word has 8 or more characters and “weak” otherwise

I have two variables, *gender* and *pregnant*. The user has input values for these variables. Gender contains the string “male” or “female”, pregnant contains the value True or False

Set 3

Write a code fragment to

1. Print the gender of the user
2. Print “check pregnant” if the gender is female
3. Print “Amount = 2300” if the gender is male
4. Print “Amount = 2000” if the gender is female
5. Print “Amount = 2300” if the gender is male and “Amount = 2000” if the gender is female
6. Print “Amount = 2000” if pregnant is False
7. Print “Amount = 2000” if pregnant is False and “Amount = 2200” otherwise
8. Print “Amount = 2300” if the gender is male, “Amount = 2000” if the gender is female and pregnant is false and “Amount = 2200” otherwise

Set 4

I have two variables, *customer_type* and *total_price*. Customer_type stores the string “standard”, “priority” or “premium”. Price stores the total price of an order, as a decimal number, e.g. 342.78

Write a code fragment to

1. Print the price with a 5% discount
2. Print the price with a 10% discount
3. Print the price with VAT added at 20%
4. Print the price with an extra £10 shipping charge
5. Print the price with an extra £10 shipping charge then VAT at 20%
6. Print the price with a 5% discount, then a £10 shipping charge then VAT at 20%
7. Print the price if they are a standard customer, and print the price with a 5% discount otherwise
8. Print the price if they are a standard customer, print the price with a 5% discount if they are a priority customer and the price with a 10% discount if they are a premium customer
9. Print the price + VAT if they are a standard customer, print the price with a 5% discount +VAT if they are a priority customer and the price with a 10% discount +VAT if they are a premium customer.
10. Print the price according to the following conditions
    1. Standard customers pay full price and £10 shipping
    2. Priority customers get a 5% discount and £10 shipping
    3. Premium customers get 10% discount and free shipping

    All customers pay VAT on the total after discounts and shipping

Count-controlled iteration

Set 1

Here is a list called *items*

items = [“Apple”, “Banana”, “Cherry”, “Date”, “Eggfruit”, “Fig”]

Write a code fragment to

1. Print every item in the list
2. Print every item in the list backwards
3. Print every other item in the list
4. Print every other item in the list backwards
5. Print every item in the list that has more than 5 characters
6. Print every item in the list that has exactly 5 characters
7. Print every item in the list that has 5 or fewer characters
8. Print the first three characters of each item
9. Print each item in capitals
10. Print each item in Sentence Case
11. Print the first three characters of each item in capitals
12. Print the first three characters of each item which has more than 5 characters

Set 2

I have an array of prices for some items

prices = [2.38, 5.12, 3.12, 15.88, 6.49, 2.10, 7.2, 8.55, 12.5, 0.55, 6.22]

Write a code fragment to

1. Print all the prices
2. Print all the prices backwards
3. Print every other price in the list

4. Print all the prices over £7
5. Print all the prices with 10% discount
6. Print all the prices with 20% VAT added
7. Print all the prices; prices over £10 get a 10% discount
8. Print all the prices; prices over £10 get a 10% discount and VAT added at 20%, other prices only get VAT added

9. Print the total price
10. Print the number of prices in the list
11. Print the average price
12. If the total price is over £10, print the total after a 10% discount, otherwise print the total price
13. If the total price is over £10, print the total after a 10% discount with 20% VAT added, otherwise print the total price with 20% VAT added, but no discount

14. Print each price that is greater than the average price

Condition-controlled iteration

Set 1

Write a code fragment to

1. Ask the user to input a string. The string should be stored into a variable called word
2. Ask the user to input a string and store it in a variable called word. Print true if the input has 8 or more characters
3. Ask the user to input a string and store it in a variable called word. Print true if the input starts with an "A"

4. Ask the user to input a string and store it in a variable called word. Keep asking until they input a word with 8 or more characters
5. Ask the user to input a string and store it in a variable called word. Keep asking until they input a word with 12 or more characters
6. Ask the user to input a string and store it in a variable called word. Keep asking until they input a word with less than 5
7. Ask the user to input a string and store it in a variable called word. Keep asking until they input a word that starts with an "A"
8. Ask the user to input a string and store it in a variable called word. Keep asking until they input a word that starts with an "!"
9. Ask the user to input a string and store it in a variable called word. Keep asking until they input a word with the same first and last letters

10. Ask the user to input a string and store it in a variable called word. Keep asking until they input a word with 8 or more characters which starts with an "A"
11. Ask the user to input a string and store it in a variable called word. Keep asking until they input a word with 8 or more characters which starts with an "M" and ends with an "!"

Set 2

Write a code fragment to

1. Ask the user to input a number, and store it in a variable called hours
2. Ask the user to input a number, and store it in a variable called hours. Keep asking until they input a number greater than 35
3. Ask the user to input a number, and store it in a variable called hours. Keep asking until they input a number less than 50
4. Ask the user to input a number, and store it in a variable called hours. Keep asking until they input a number between 35 and 50

Set 3

Write a code fragment to

1. Ask the user to input two numbers and store them in variables a and b
2. Ask the user to input two numbers and store them in variables a and b. Keep asking until a is greater than b
3. Ask the user to input two numbers and store them in variables a and b. Keep asking until a is less than b
4. Ask the user to input two numbers and store them in variables a and b. Keep asking until a is equal to

A standard algorithm

I have two variables, a and b. Each contains an integer.

Write a code fragment to

1. Print true if a is equal to b
2. Print true if a is greater than b
3. Print true if a is less than or equal to b
4. Print true if a is greater than b and false otherwise
5. Print “a” if a is greater than b and “b” otherwise

6. Swap the two variables
7. Swap the variables if a is greater than b
8. Swap the variables if a is greater than b and print “swapped”

9. Make both variables equal to the larger of the two numbers
10. Make both variables equal to the smaller of the two numbers

I have a list called numbers, containing a list of integers for example, numbers = [3, 7, 5, 1, 8, 2, 9, 4]

The length of the list is length(numbers)

Write a code fragment, using a count-controlled loop and the list index, to go through the list and

1. Print every number in the list
2. Print every number in the list backwards
3. Print every number except the last one
4. Print every number except the first one
5. Print each number in the list and the one after it, e.g. 3,7   7,5
6. Print each number in the list and the one before it e.g. 7,3   5,7
7. Print the numbers in pairs, working in from the ends, e.g. 3,4   7, 9

8. Add one to every number in the list and overwrite the number in the list
9. Double every number in the list and overwrite the number in the list
10. Replace each number with the number before it in the list. The first number becomes 0
11. Replace each number with the number after it in the list. The last number becomes 0

12. Check each number - print true if the number is greater than 5
13. Check each number - print true if the number if greater than 5 and false otherwise
14. Check each number - print each number that is greater than 5
15. Check each number - print true if the number is greater than the next number
16. Check each number - print the number if it is greater than the next one

17. Overwrite the number with 5 if it is greater than 5
18. Overwrite the number with 1 if it is greater than 5 or 0 otherwise
19. Check each number – count how many are greater than 5
20. Check each number – print true *once* at the end if any are greater than 5
21. Check each number - print true if the number is greater than the next number and count how many times this happened
22. Check each number - print true *once* at the end if any number is greater than the next number
23. Compare each number to the next one. If the first number is bigger, swap the numbers
24. Compare each number to the next one. If the first number is bigger, swap the numbers and print "swapped"
25. Compare each number to the next one. If the first number is bigger, swap the numbers and count how many times this happened
26. Compare each number to the next one. If the first number is bigger, swap the numbers and print "swapped" *once* at the end if any swaps occurred

27. Repeat the following code fragment length(numbers) times
    a. Go through the whole list and compare each number to the next one. If the first number is bigger, swap the numbers and print "swapped" *once* at the end if any swaps occurred
28. Repeat the following code fragment until no swaps occurred
    a. Go through the whole list and compare each number to the next one. If the first number is bigger, swap the numbers

2D arrays

I have a 2D array called *scores* that can be visualised like this

3	5	6	1	4	6	5	8
2	7	3	4	2	0	5	9
9	9	6	7	9	7	8	9
4	5	7	4	8	9	5	7
7	4	8	4	7	5	6	8
6	2	6	8	5	4	7	6

The numbers represent scores out of 10 for 6 different students on class tests, i.e. student 1 scored 3, 5, 6, 1, 4, 6, 5, 8 across their tests.

The array is indexed from 0. So scores[0, 0] is 3, score[1,0] is 2 and scores[0,1] is 5

Write a code fragment to

1. Print all the scores for the first student
2. Print all the scores from the last student
3. Print all the scores from test 1
4. Print all the scores from test 3
5. Print all the scores from the last test
6. Print the total score for the first student
7. Print the average score for the first test

8. Print all the scores by student. The scores for each student should be on a separate line
9. Print all the scores by test. The scores for each test should be on a separate line

10. Print the total score for each student
11. Print the average score for each student
12. Print the average score for each test
13. Print the average score for all the tests

14. Print the number of tests where the first student scored greater than 5
15. Print the number of students who scored greater than 5 on the first test
16. Print the total number of scores greater than 5 over all the tests
17. For each student, print the number of tests where they scored greater than 5
18. For each test, print the number of students who scored greater than 5

19. Print the number of any student who scored greater than 5 on all their tests
20. Print the number of any test where all the students scored greater than 5

21. Print the number of the first student to score 9 on any test, then stop
22. Print the number of the first student to score 9 on any test and which test it was, then stop
23. Print the number of the first test on which any student scored 0, then stop
24. Print the number of the first test on which any student scored 0 and which student it was, then stop
25. Print the average score of the first student to score an average greater than 5

Drills and Skills – answers to selected question (the last problem from most sets)

Selection Statements

Set 1

Q6.

```
if score > 90 then
    print “distinction”
elseif score > 75 then
    print “merit”
elseif score > 50 then
    print “pass”
else
    print “fail”
endif
```

Set 2

Q5.

```
if length(word) >= 12 then
    print “strong”
elseif length(word) >= 8 then
    print “good”
else
    print “weak”
endif
```

Set 3

Q8.

```
if gender == “male” then
    print “Amount = 2200”
else
    if pregnant == True then
        print “Amount = 2200”
    else
        print “Amount = 2000”
    endif
endif
```

Note that this solution could be made more compact – it is left as a nested if to show the two cases more clearly to students. How and why this can be reduced may be a good discussion for students.

Set 4

Q10.

```
if customer_type == “premium” then
    total_price = total_price * 0.9
elseif customer_type = “priority” then
    total_price = total_price * 0.95 + 10
else
    total_price = total_price + 10

total_price = total_price * 1.2    //add VAT for all customers
print total_price
```

Count-controlled iteration

Set 1

Q12

```
for i = 0 to length(items) - 1
    item = items[i]   //item placed in its own variable to make the indexes clearer later
    if length(item) > 5 then
        print item[1] + item[2] + item[3]
    endif
next i
```

Set 2

Q14.

```
total = 0
for i = 0 to length(prices) – 1
    total = total + prices[i]
next i
average = total / length(prices)

for i = 1 to length(prices) – 1
    if prices[i] > average then
        print prices[i]
    endif
next i
```

Condition-controlled iteration

Set 1

Q11.

```
valid = False
while valid == False
    word  = input("Please enter a string")
    if length(word) >= 8 and word[0] == "M" and word[length(word) – 1] == "!" then
        valid = True
    endif
endwhile
```

A standard algorithm

Q28. This is bubble sort!

2D Arrays

```
average = 0
while average <= 5
    for i = 0 to 5
        total = 0
        for j = 0 to 7
            total = total + scores[i, j]
        next j
        average = total / 8
endwhile
print average
```