YOU MUST WORK ON THIS SERIES OF EXERCISES (SHORT 1-8 and the MINIPROJECT) AND GET THEM MARKED IN ONLINE VIVAS WEEK-BY-WEEK.

Grade F- -: Short Assessed Programming Exercise 1: Output

To pass this exercise you must demonstrate that you are able to do the following in solving the problem

- ❑ **EXPLAIN HOW YOUR PROGRAM WORKS**
- ❑ **Write simple code that compiles and runs in JHUB**
- ❑ **Write code that correctly uses output instructions**
- ❑ **Write code split in to methods**
- ❑ **Include simple comments – at least the actual author's name and date**
- ❑ **Write a literate version of the code**

When your program works, test it and check it ticks all the above boxes then have it assessed by a demonstrator. If you are having problems or don't understand the requirements ask for help.

Big Initials Write a program to print out your initials down the page in block letters using the same letter to draw it out of. A blank line should separate the initials. Each initial should be printed by a separate method. For example, my initials are PC. My program should therefore print:

```
PPPPP
P   P
PPPPP
P
P

CCCCC
C
C
C
CCCCC
```

HINT: Plan what your output will look like before starting to write the program

You can modify one of the programs from the course QM+ page or interactive notebooks rather than writing it out from scratch.

TESTING Short Assessed Programming Exercise 1: Output

You should check as a minimum that it meets the tick boxes at the top of the page and:

- ❑ **The program compiles correctly**
- ❑ **The program runs correctly**
- ❑ **It prints EXACTLY the right thing (reread the above description and make sure you didn't miss anything)**
- ❑ **Double check all comments make sense for THIS program**

As this program just does output you should check the output carefully – does it show both your first and last initial (at least). Are the correct letters used to form the shapes? Is there a blank line between the letters?

Remember, programming can be tricky at first if you are a novice, but …
you are capable of learning to program if you keep at it. Ask for help if you need it. The more programs (however small and simple) you write the easier it gets. Doing the extra programming exercises from the interactive notebooks/workbook will help as they take you through the concepts gradually.

Grade F-: Short Assessed Programming Exercise 2, Input, Calculation and Variables

To pass this exercise you must demonstrate that you are able to do the following in solving the problem

- ❑ **Write a literate version of the code**
- ❑ **EXPLAIN HOW YOUR PROGRAM WORKS**
- ❑ **Write a program that uses input instructions**
- ❑ **Write a program that uses variables**
- ❑ **Write a program that uses final variables**
- ❑ **Write a program that uses arithmetic expressions**
- ❑ **Write a program that is split in to methods at least one of which returns a result**
- ❑ **Write a program that makes simple use of comments – at least the author's name and date and explanation of what the program does overall**

When your program works, test it and check it ticks all the above boxes then have it assessed by a demonstrator. If you are having problems or don't understand the requirements ask for help. .

Ordering Enough Balloons

Write a program for an artist who is creating art instillation "experiences" consisting of rooms totally full of balloons that people will walk around in. Different instillations will be in different sized rooms and different sized balloons so need different numbers of balloons to fill them. The salesperson will enter the length, width and height of the room in centimetres. Write separate methods to input each of these values (they may possibly call other general methods) and return the results. These three numbers are multiplied together to get the volume of the room. This should by then be converted to m3 (calculated by dividing the volume by 1,000,000). This is then divided by the volume taken up by each balloon in m3 as given by the user. Print out the final number of balloons to be ordered along with the intermediate results as in the examples below. The final answer should be given as an integer (rounded down) as below.

An example run of the program is as follows (numbers in bold are typed in by the user, either pop-up boxes or the console may be used for input and output):

```
Length of the room (in cm)? 4600
Width of the room (in cm)? 310
Height of the room (in cm)? 200
What is the balloon volume (in m3)? 0.03
Your room volume is 285.2 m3.
You need 9506 balloons.
```

Another example run:

```
Length of the room (in cm)? 1000
Width of the room (in cm)? 520
Height of the room (in cm)? 330
What is the balloon volume (in m3)? 0.07
Your room volume is 171.6 m3.
You need 2451 balloons.
```

HINT: You can round down to an integer, getting rid of spurious decimal places, using the integer cast operator (int) - see the interactive notebook on types.

Take an example program from the QM+ page or interactive programs as your starting program – modifying it to do this.

TESTING Short Assessed Programming Exercise 2, Input, Calculation and Variables

You should check as a minimum that it meets the tick boxes at the top of the page and:

- ❑ **The program runs correctly for all input (not just he examples above) including extreme values.**
- ❑ **The program should clearly indicate what should be typed when the user is needed to enter values (including indicating values not acceptable).**

This program is now more complicated. You cannot just run it once and see if it works as what it does depends on the values input. You will need to run it lots of times – enough to convince yourself it always works. What about extreme values – very big or very small? Zero is always a good value to test for. For this exercise, if the program crashes when bad values are entered that is ok as long as the user was warned in some way not to type those values beforehand. Check the correct answer is given. Check carefully there are no missing spaces or punctuation in the output. Inspecting your code by eye, including checking all variables are given a value before the value is used, is always a good idea. Check the comments make sense.

Grade F: Short Assessed Programming Exercise 3: Making Decisions

To pass this exercise you must demonstrate that you are able to do the following in solving the problem

- ❑ **Write a literate version of the code**
- ❑ **EXPLAIN HOW YOUR PROGRAM WORKS**
- ❑ **Write a program that uses if-then-else statements.**
- ❑ **Write a program that is split in to methods and includes methods that return a result.**
- ❑ **Write a program that includes useful comments, at least one per method saying what it does.**
- ❑ **Write a program that uses indentation in a way that makes its structure clear.**

When your program works, test it and check it ticks all the above boxes then have it assessed by a demonstrator. If you are having problems or don't understand the requirements ask for help.

Parking Charges Write a program that works out the amount a person has to pay for parking in a car park in a tourist town. If they say they are disabled, they are told it is free. Otherwise they enter the number of hours as a whole number (1-8) that they wish to park as well as whether they have an "I live locally" badge or are an old age pensioner both of which leads to a discount. The program tells them the cost to park.

The calculation is done in this program as follows. If they are disabled it is free. Otherwise … Take number of hours they wish to park and give a basic charge:

- 1 hour: 3.00 pounds
- 2-4 hours: 4.00 pounds
- 5-6 hours: 4.50 pounds
- 7-8 hours: 5.50 pounds

Next modify the resulting charge based on whether they are local or not:

- If local: subtract 1 pound
- If OAP: subtract 2 pounds

Your program MUST include methods including ones that

- asks for the hours and returns the basic charge.
- asks whether they live locally / are an OAP and returns the amount to subtract

An example run of the program (words in **bold** are typed by the user: pop-up boxes may be used for I/O):

```
Are you disabled? Yes
Parking for you is free
```

Another example run:

```
Are you disabled? No
How many hours do you wish to park (1-8)? 5
Do you have an "I live locally badge"? Yes
Are you an OAP? Yes
The parking charge for you is 1.50 pounds.
```

Another example run:

```
Are you disabled? No
How many hours do you wish to park (1-8)? 1
Do you have an "I live locally badge"? Yes
Are you an OAP? No
The parking charge for you is 2 pounds.
```

TESTING Short Assessed Programming Exercise 3: Making Decisions

You should check as a minimum that it meets the tick boxes at the top of the page and:

- ❑ **The program runs correctly for all input.**
- ❑ **All methods individually work correctly / return the correct result**
- ❑ **All branches of the IF-THEN-ELSE statement work**
- ❑ **The program deals with input it doesn't know about**

Decision statements add a new level of difficulty for testing. You need to make sure every line works, but when you run the code some of the lines are not executed – as in any if statement either the then case or the else case is executed not both. You must test the program by running it lots of times with different values, choosing values that will definitely test all the lines of code (**so test all branches**) between them. Also remember to check carefully the output is right (spaces, punctuation, etc). By breaking the program in to methods you can test each method separately (as you write it). Testing is easier if you write a test method adding methods as you write them. It can be called from main but commented out of the final version

It is a good idea to inspect the code by eye too. This is called doing a "Programmer's Walkthrough", "tracing the code" or "dry running". Does it appear to do the right thing stepping through the execution on paper?

Grade D: Short Assessed Programming Exercise 4: Records and For Loops

To pass this exercise you must demonstrate that you are able to do the following in solving the problem

- ❑ **Write a literate version of the code**
- ❑ **EXPLAIN HOW YOUR PROGRAM WORKS**
- ❑ **Write a program using counter controlled FOR loop statements.**
- ❑ **Write a program that creates user-defined types, defining and using records.**
- ❑ **Write a program that has at least one method that take argument(s) and returns a result**
- ❑ **Write a program that includes useful comments, at least one per method saying what it does.**
- ❑ **Write a program that uses indentation in a way that makes its structure clear.**
- ❑ **Write a program that uses variable names that give an indication of their use.**

When your program works, test it and check it ticks all the above boxes then have it assessed by a demonstrator. If you are having problems or don't understand the requirements ask for help.

Tourist Attraction Information Write a program that gives information about tourist attractions. The user should first input how many tourist attractions they wish to ask about and then be allowed to name that many places. The program should give their opening time (assumed on the hour and in the morning for the purposes of this exercise) and whether they open on bank holidays. A new type called Attraction must be created (a record type) and each separate piece of information about an attraction should be stored in a separate field of the record (its name - a String, opening time - an integer, closing time - an integer, and whether they open on bank holidays -a boolean).

A separate method must be written that given a String (an attraction name) as argument returns a String containing the correct information about the attraction to print. The String should then be printed by the calling method. An example run of the program (**bold** words are typed by the user): Your answer need only include the information about known stations as in this example.

```
How many attractions do you need to know about? 4

Name attraction 1? The Eden Project
The Eden Project opens on bank holidays.
It opens at 9am.

Name attraction 2? Tate Modern
Tate Modern does not open on bank holidays.
It opens at 10am.

Name attraction 3? The Zoo
I have no information about that attraction.

Name attraction 4? London Zoo
London Zoo opens on bank holidays.
It opens at 10am.
```

TESTING Short Assessed Programming Exercise 4: For Loops and Records

You should check as a minimum that it meets the tick boxes at the top of the page and:

- ❑ **The program runs correctly for all input.**
- ❑ **All methods return the correct result**
- ❑ **The program always does the correct number of iterations**
- ❑ **All FIELDS of any RECORD are set and accessed correctly**
- ❑ **The program deals with input it doesn't know about**
- ❑ **Running totals are correct at all points through the loop**

For programs with records, you need to be sure the fields are both set and accessed correctly. Testing is easier if you write a special test method adding tests for new methods as you write them. As with the *if* statements, loops execute code depending on a test. You need to check that the loop does execute exactly the right number of times every time. Doing dry run/programmer's walkthroughs can be especially important. Make sure loops can't run forever for any input (including input the programmer didn't think of the user ever entering such as empty strings, zero and negative numbers). A neat little debugging hack is to stick print statements in the code, so that when you run it, you get some feedback on what the code is doing. Perhaps "This program waz ere" or even "This is the i^{th} time through this loop", etc..

Grade C: Short Assessed Programming Exercise 5: Arrays

To pass this exercise you must demonstrate that you are able to do the following in solving the problem

- ❑ **Write a literate version of the code**
- ❑ **EXPLAIN HOW YOUR PROGRAM WORKS**
- ❑ **Write a program that uses an array manipulated in a loop.**
- ❑ **Write a program that includes methods that take multiple arguments including array arguments.**
- ❑ **Write a program that is well indented.**
- ❑ **Write a program that contains helpful comments.**
- ❑ **Write a program that uses variable names that give an indication of their use.**
- ❑ **Use final variables to store literal values rather than them appearing through the code.**
- ❑ **Ensure all variables have minimal scope**

When your program works, test it and check it ticks all the above boxes then have it assessed by a demonstrator. If you are having problems or don't understand the requirements ask for help.

Top Olympic/Paralympic Medal winners Write a program that uses a for loop to ask the user to name 5 Olympians or Paralympians. They should say how many medals each won and what their sport is. Their names should be kept in one array, the number of medals in a second array in the corresponding position, and their sport in a third. A final variable should be used to store the array size.

The program should then give the average number of medals these people won (rounded to the nearest integer). It should finally print out each person's information in a list in reverse order, with commas separating the sport, name, medals won (suitable for input to a spreadsheet program).

```
Name Olympians/Paralympian 1? Kadeena Cox
How many medals did he/she win? 6
What sport did he/she compete in? Running/Cycling

Name Olympians/Paralympian 2? Laura Kenny
How many medals did he/she win? 6
What sport did he/she compete in? Cycling

Name Olympians/Paralympian 3? Mo Farah
How many medals did he/she win? 4
What sport did he/she compete in? Athletics

Name Olympians/Paralympian 4? Sarah Storey
How many medals did he/she win? 28
What sport did he/she compete in? Cycling/Swimming

Name Olympians/Paralympian 5? Adam Peaty
How many medals did he/she win? 5
What sport did he/she compete in? Swimming
Between them they won an average of 10 medals each.

Swimming, Adam Peaty, 5
Cycling/Swimming, Sarah Storey, 28
Athletics, Mo Farah, 4
Cycling, Laura Kenny, 6
Running/Cycling, Kadeena Cox, 6
```

HINT: Have a variable keep a running total. Round a number to the nearest int using the Math.rint() method.

TESTING Short Assessed Programming Exercise 5: Arrays

You should check as a minimum that it meets the tick boxes at the top of the page and:

- ❑ **The program runs correctly for all input.**
- ❑ **No out of bound errors through the way the loop processes the array**

Arrays are a common source of errors. The points about loops apply, but it's always worth checking the code cannot run off the end of the array (make sure it can not visit non-existent position 5 in an array of length 5 for example – remember it starts counting positions from 0!) Checking position 0 is used correctly is important too.

Grade B: Short Assessed Programming Exercise 6: While Loops

To pass this exercise you must demonstrate that you are able to do the following in solving the problem

- ❑ **Write a literate version of the code**
- ❑ **EXPLAIN HOW YOUR PROGRAM WORKS**
- ❑ **Write a program using WHILE loops.**
- ❑ **Use indentation well.**
- ❑ **Provide helpful comments in the code.**
- ❑ **Use variable names that give an indication of their use.**
- ❑ **Ensure all variables have minimal scope**

When your program works, test it and check it ticks all the above boxes then have it assessed by a demonstrator. If you are having problems or don't understand the requirements ask for help.

Train Count Write a program that is to be used to investigate how late trains passing through a given station are. The trains are so dire that virtually all trains seem to be running late. It should use a while loop to repeatedly ask the user to name the destination of the train that just departed. It should stop when the special code XXX is entered, and then give the total minutes late of all trains departed and which train was most punctual (ie least late). For example, one run might be as follows.

```
What is the destination of the train that just departed? Liverpool
How many minutes late was it? 7

What is the destination of the train that just departed? Newcastle
How many minutes late was it? 1

What is the destination of the train that just departed? Southampton
How many minutes late was it? 5

What is the destination of the train that just departed? Taunton
How many minutes late was it? 3

What is the destination of the train that just departed? XXXX

The trains were in total 16 minutes late.
The most punctual train was to Newcastle. It was 1 minute late.
```

HINT: Have variables that store the most punctual train seen so far and its destination. If more than one train is equally punctual you may choose either.

Make sure you comment your program with comments that give useful information, use indentation consistently and that your variable names convey useful information.

TESTING Short Assessed Programming Exercise 6: While Loops

You should check as a minimum that it meets the tick boxes at the top of the page and:

- ❑ **The program runs correctly for all input.**
- ❑ **The loop is entered correctly the first time.**
- ❑ **The loop terminates correctly even if terminated immediately.**

When the number of times round the loop can vary, use test input values that check it for different numbers of iterations (times round the loop). Odd cases to check are programmers messing up so the loop body always runs just once, or no times at all. Make sure it works if the user ends the program immediately. Doing dry run/programmer's walkthroughs can be especially important. Make sure loops can't run forever for any input including odd values input.

You can see what is happening inside a loop by adding an extra print statement in the body of the loop eg System.out,println("IN THE LOOP") so you can see it is entering the loop. The number of times that message is printed also shows you how many times the loop body repeats. Remember to delete or comment out such test statements from the final version!

Remember, programming can be tricky at first, but …
you are capable of learning to program if you keep at it.

Grade A: Short Assessed Programming Exercise 7: Accessor Methods

To pass this exercise you must demonstrate that you are able to do the following in solving the problem

- ❑ **Write a literate version of the code**
- ❑ **EXPLAIN HOW YOUR PROGRAM WORKS**
- ❑ **Write programs consisting of multiple methods, call them with multiple arguments and return values from them**
- ❑ **Define an abstract data type with record fields accessed ONLY by procedural programming style accessor methods and an initialisation method all defined in the class containing main**
- ❑ **Write a program that is well indented.**
- ❑ **Write a program with each method individually and helpfully commented on its use.**
- ❑ **Write a program that uses variable names that give a clear indication of their use.**
- ❑ **Write a program where all variables have minimal scope**

When your program works, test it and check it ticks all the above boxes then have it assessed by a demonstrator. If you are having problems or don't understand the requirements ask for help.

Paralympic Relay A new paralympic relay competitions involves competitors of different disability classes making up a team. The first leg must have someone from T11 or T13, leg two has someone from T61 or T62, leg three has someone from class T35 or T36 and the final leg has someone from the T51 or T52 class.

Write a program that checks the legality of relay teams given the country and the disability class of the entrant for each relay leg. A new type called **UniversalRelayTeam** should be created (a record type). Information about a team should be stored in fields of the record: (country - a String, Leg 1, Leg 2, Leg 3 and Leg 4 storing the disability classes of each person as 4 integers). This record **must** be defined in **a new class containing NOTHING but the field definitions (no methods at all in the record class)**.

In the main class (the one containing ALL methods including the main method), an initialisation method should be provided to create records. It should take as arguments the values previously input about a team. Accessor methods must also be defined for the fields in the main class (they should NOT be in the UniversalRelayTeam class for this exercise**).**

No other method should access the record fields directly apart from the accessor methods.

An example run of the program (words in **bold** are typed in by the user and pop-up boxes may be used for input and output): The following is an example run of the program:

```
What country is the team representing? GB
What is the disability class for leg 1? T 11
What is the disability class for leg 2? T 61
What is the disability class for leg 3? T 52
What is the disability class for leg 4? T 12
The GB team is: Leg 1, T11; Leg 2, T61; Leg 3, T52; Leg 4, T12
Leg 3 (T52) is not legal.
Leg 4 (T12) is not legal.
```

TESTING Short Assessed Programming Exercise 7: Accessor methods

You should check as a minimum that it meets the tick boxes at the top of the page and:

- ❑ **The program runs correctly for all input.**
- ❑ **Contains multiple methods including accessor methods that each individually work and return the correct results for a range of inputs.**

Methods make the tester's job easier – a reason for using them! You can test each method separately – make sure it works as it should before worrying about whether the program as a whole does. If methods work then later errors found will be in the code that uses them not the methods. Create a test plan method called from main but commented away in the final version that tests each method printing results returned checking they are correct. Once you are sure accessor methods work, you can ignore how the record is implemented.

Grade A+: Short Assessed Programming Exercise 8: Recursion

To pass this exercise you must demonstrate that you are able to do the following in solving the problem.

- ❑ **Write a literate version of the code**
- ❑ **EXPLAIN HOW YOUR PROGRAM WORKS**
- ❑ **Write a program that uses recursion in an appropriate way to solve a recursive problem.**
- ❑ **Write a program that is well indented, to clearly show the program structure.**
- ❑ **Write a program that contains helpful comments with ALL methods clearly commented.**
- ❑ **Write a program that uses variable names that give a clear indication of their use.**
- ❑ **Write a program where all variables have minimal scope**

When your program works, test it and check it ticks all the above boxes then have it assessed by a demonstrator. If you are having problems or don't understand the requirements ask for help.

Recursive Parsing Calculator

HINT: Before attempting this exercise, see the simplerecursiveparser.java in the example programs of the recursion unit on QM+ and the related booklet about Language, grammars and recursion that explains it.

Write a program that recursively parses expressions, input as strings, from the following recursively defined language and calculates and prints out the answer to the calculations. Legal expressions in this language involve putting the operator before its arguments (this is called Polish notation).

<EXP> =	* <DIGIT> <EXP> | T <EXP> | <DIGIT>
<DIGIT> =	0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C

Instead of writing 3*2, in this language you write *32 (which evaluates to 6). EXP stands for expressions. * means multiply the two digits that follow (after evaluation) so *24 is 8 and *5*34 is 60 as it first multiplies 3 and 4 to get answer 12 then multiplies 5 and 12 to get 60. Tn means add the next 3 numbers from n (so eg T3 means 3+4+5=12, T7 means 7+8+9=24), DIGIT gives a way to express numbers up to 12 as a single digit (so A means 10, B means 11, C means 12).

Only single digits are allowed and spaces are not allowed. Further legal expressions can be seen below.

An example run of the program (characters in **bold** are typed in by the user and pop-up boxes may be used for input and output):

```
Please input the expression B
The answer is 11
```

Another example run:

```
Please input the expression T4
The answer is 15
```

Another example run:

```
Please input the expression T*23
The answer is 21
```

Another example run:

```
Please input the expression T*2*1T2
The answer is 57
```

Another example run:

```
Please input the expression *2*3*A5
The answer is 300
```

Make sure you split the program in to multiple methods, and use a series of recursive methods following the structure of the recursive definition about expressions. You must **not** use an explicit loop at all in your program. Comment your program with useful comments that give useful information, with every method commented with what it does, use indentation consistently and ensure that your variable names convey useful information. Your variables should be positioned so that their scope is small.

TESTING Short Assessed Programming Exercise 8: Recursion

You should check as a minimum that it meets the tick boxes at the top of the page and:

- ❑ **The program runs correctly for all input.**
- ❑ **Each method individually works**
- ❑ **Each recursive method works for both base cases and step cases.**
- ❑ **The program gives a suitable message for different kinds of invalid input**