

Name: Ahmad Ibrahim

ID:22110139

Tutor Name: Samer Suleiman

Course Name: Programming

Part 1:

Task 1 of part 1(Report)

1.1.1:

Outline the steps involved in the process of building an application from writing to execution and describe how you used these steps to build your system.

1- Identify the problem (requirements gathering):

Build an age calculator and store the result in an array.

The elements must be sorted in descending order after converting the age to seconds, you need to take the size of the array as the number of persons to compare. The system must be accurate and work perfectly, the system accepts the input as day, month, year.

2- Find a solution and design the architecture: to find a solution and specify the equations you need to use its preferably to write a pseudo code or produce a flow chart to make easier, in this project I used specific equations to calculate the age based on the date of birth until 13/5/2023.

3- Write your code: Coding is an essential part of the programming process. Coders should create instructions

from the solution and implement them into code for the computer.

I implemented this code using java programming language. It involved (if else) statements, for loops

Methods and other concepts.

4- Testing the code: Testing is a critical process used to determine the quality of a program, find errors and bugs. Working to find bugs and try fixing them helps understand the quality of the program.

I tested my code multiple times and there was some logical and runtime errors to fix such as the equations used to calculate the age.

5- Enhance the code: It’s important to continue enhancing the code to ensure that it meets changing requirements and stays up to date with the latest developments.

1.1.2:

In computer science, algorithm is a set of steps to accomplish a task it should also be clear and independent.

Brute Force Algorithm: It is the simplest approach for a problem. A brute force algorithm is the first solution that comes up to mind when we face a problem.

Hashing Algorithm: Hashing algorithms work likely to the searching algorithm. But it contain an index with a key ID. In hashing, a key is assigned to specific data.

Searching Algorithm: Searching algorithms are that are used to get an element or groups of elements that you are looking for from a particular data structure. There are different types based on their logic or the data structure.

Insertion sort algorithm:

1-for i = 0 to size-1 do

2-for j = i+1 to size do

3-if arr[i] > arr[j] then

4-temp = arr[i]

5-arr[i] = arr[j]

6-arr[j] = temp

7-end if

8-end for

9- end for

* Using an algorithm is more efficient than brute force approach for some reasons like:

1. Saving effort and time.
2. Scalability.
3. More accurate.

* In my code I used bubble sort to sort the elements in the array which enhanced the code and made it more efficient .

1.1.3.1: here is my pseudo code🡪

1- start

2-Print” Enter Array Size”

3- Scan integer size from the user

4-create a long array with the taken size

5- repeat steps from6-9 until size-1

6-print(enter date of birth as year month day for person+1)

7-Take input type long values for year, month, and day from the user

8- Calculate age in seconds by calling function calcAge(year, month, day)

9- Assign the ageInSec value to arr[i]

10- Calculate age in days using function calcDay(year, month, day)

11- Print the age in seconds+ age in days

12- use the method sortArrayDescending(arr) to sort the array in descending order.

13- Print (Sorted array in descending order:)

14- repeat step 15 until size-1

15- Print arr[i]

16- End

Method calcAge(year,month,day):

1. Initialize ageNsec variable with type long.
2. Subtract year from 2023 and multiply by 31536000 (365\*24\*60\*60) to calculate sum of years in seconds.
3. Calculate sum of months in seconds by subtracting month from 5 and multiplying by 30.
4. Calculate sum of days in seconds by taking the absolute value of 13-day.
5. If month is greater than or equal to 5, do steps 6-7, else do steps 8-9.
6. If day is greater than or equal to 13, assign (sumYear - sumMonth - sumDay) to ageNsec, else assign (sumYear - sumMonth + sumDay) to ageNsec.
7. Return ageNsec.
8. If day is greater than or equal to 13, assign (sumYear+ sumMonth- sumDay) to ageNsec, else assign (sumYear + sumMonth + sumDay) to ageNsec.
9. Return ageNsec.

Method calcDay(year,month,day):

1. initialize agenda.
2. Calculate days in years by using this equation (2023-year)\*365 c
3. Calculate days in months by using this equation (5-months)\*30 and store it in monthNday.
4. Take the absolute value from (13-days) to calculate days and store it in days
5. If months greater or equal 5 do steps 6,7 else do 8,9
6. If day is greater than or equal to 13, store (yearNday - monthNday - days) to ageNday, else store (yearNday - monthNday + days) to agenda
7. Return ageNday
8. If day is greater than or equal to 13, store (yearNday + monthNday - days) to ageNday, else store (yearNday + monthNday + days) to ageNday
9. Return ageNday

1.1.3.2: I used the Bubble sort algorithm to sort the array

public static void sortArrayDescending(long[] arr) {

        int n = arr.length;

        for (int i = 0; i < n - 1; i++) {

            for (int j = 0; j < n - i - 1; j++) {

                if (arr[j] < arr[j + 1]) {

*// swap arr[j] and arr[j+1]*

                    long temp = arr[j];

                    arr[j] = arr[j + 1];

                    arr[j + 1] = temp;

                }

            }

        }

1.1.3.3:

It is a simple sorting algorithm that repeatedly steps through the list element by element, it then compares the current element with the one after it, swapping their values if needed (applies the condition). These are repeated until no swaps have to be performed during a pass, which means that the array is fully sorted and there is no swaps have to be performed.

This algorithm is not suitable for large data sets as its average and worst-case time complexity is quite high.

Example: arr [] = {9,11,1,5}

It starts by comparing the first two elements and doesn’t swap since 9<11. (9,11,1,5)🡪remains the same

Swap(11,1)since 1<11🡪(9,1,11,5)

Swap(11,5)since 5<11🡪(9,1,5,11)

Now, the second iteration (9,1)🡪(1,9,5,11)

(9,5)🡪(1,5,9,11)

Now, the array is already sorted, but our algorithm does not know if it is completed.

The algorithm needs one whole pass without any swap to know it is sorted.

( 1,5,9,11 ) –> ( 1,5,9,11 ), No change as 1<5

1.1.3.4:

Clear and unambiguous

Well-defined inputs🡪 the inputs indicates about their function.

well-defined outputs🡪the outputs are the same as required in the task.

Finite-ness🡪there is no infinite loop in the code.

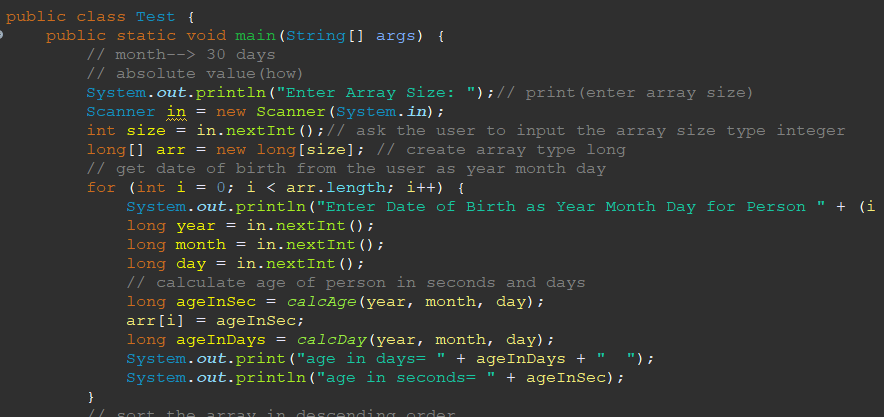
Language independent🡪 the algorithm can be represented in any programming language.

1.1.4:

Using an IDE is easier and more efficient than not using an IDE like notepad for reasons such as:

1. Running your program in an IDE from the running options is more easier than running in the terminal by writing a long command.



1. IDE has many features such as syntax error detection which make coding easier. 
2. An IDE like eclipse and others have tools that saves effort and time such as generating constructors.
3. Most of the IDE has color coding feature which indicates the use of each part of the code unlike notepad.. 5- Debugging: IDEs typically offer a built-in debugger to help find and fix errors in code. Unlike notepad which doesn’t have this feature.

6- An IDE usually has the feature to orient the code automatically.

Part 2:

Task1 of Part2

2.1.1:

Programming paradigm:

A programming paradigm is a programming style. It doesn’t necessarily relate to a specific language, but rather to the way and approach of your programming, it contains a set of concepts and strategies to assist the problem-solving process.

There are multiple types of programming paradigms:

1- imperative programming paradigm🡪 in this paradigm you give the computer some tasks to complete, then it completes them at the same time.

2- procedural programming🡪 this type of paradigm operates on a sequenced series of commands and procedures. It uses arrays, loops, if statements, variables, and functions.

3- object-oriented programming🡪 the main definition of OOP relies on multiple concepts:

a- Objects: objects are instances of classes they have state represented by attributes and actions represented by methods.

b- Classes: you can imagine it as a frame that includes the attributes and methods of the objects.

c-Inheritance: it is used when you have common attributes between classes, it allows the user to inherit the sub class from the super class, which saves a lot of time and effort.

d-polymorphism: The definition of polymorphism is related to overloading and overriding concepts.

Overloading: It’s a method that allows multiple methods with the same name but a different parameters.

Overriding: It’s a method that can be used in both sub and super class.

e- Encapsulation: it’s hiding the internal code of an object.

4- Event driven: it reacts to events entered by user or system events, it isn’t related to any programming language.

2.1.3:

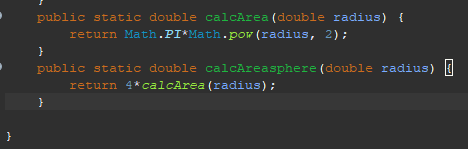
1- Procedural paradigm:

procedural paradigm follows a step-by-step command from the top all the way down, it focuses on functions and sequenced instructions. It could be suitable for simple programs, functions offers code modularity by breaking the code into smaller units, which make the code easier to fix and understand, functions also saves effort by enabling code reusability, after you define the function it can be called anywhere in the system.

When you make a static variables, they belongs to the class and not to object, which means that the variable has just one copy in the memory.

Arrays can also be created to store data which can be accessed by indexing.

Defining and calling functions examples:





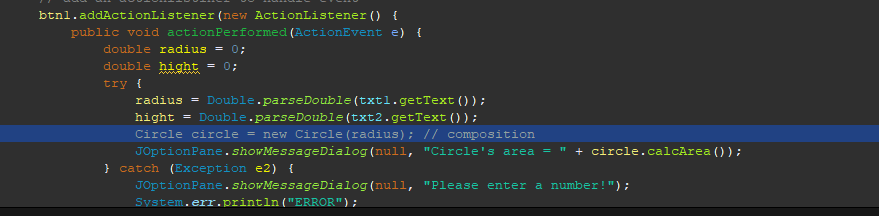
2- Object Oriented Programming:

OOP focuses on instantiate an object from a class that could be encapsulated to hide it, OOP support inheritance allowing to create a subclass from the parent,

As shown above I used private to apply abstraction, I also used Circle class as a parent class to inherit Cylinder.

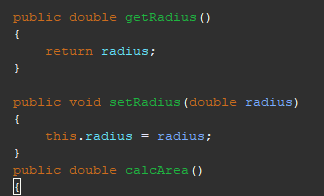
I used polymorphism overloading & overriding by creating methods with the same name but different parameters and I applied overriding by using “calcArea”

method which is defined in Circle’s class in the Cylinder class.

Composition means to have one class that contain an object from other class which I applied by instantiating a circle object in the Final class.

Encapsulation:

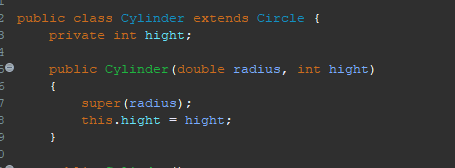




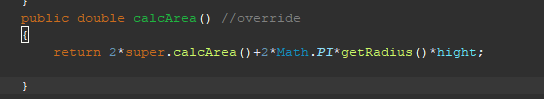
Composition: the circle object is composed of the final class.



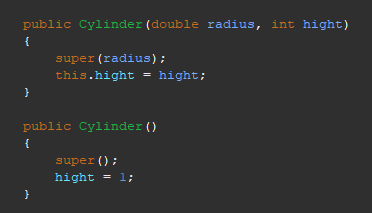
Inheritance: the subclass cylinder inherit from the superclass(circle)



Override: a subclass method override a method in the superclass



Overload: the same method with different parameters



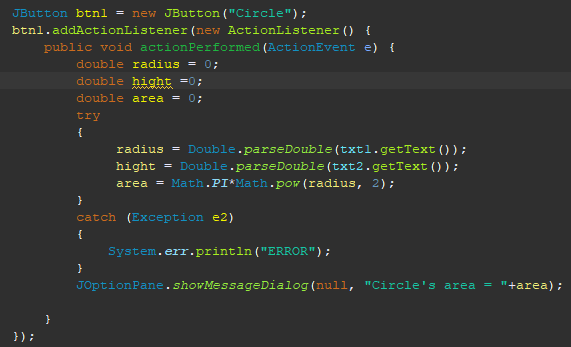
3- Event driven paradigm:

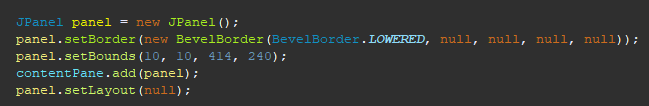
The event driven paradigm revolves around the idea of events and how they’re handled, it is often used in (GUI) graphical user interface, which enables user to interact with the computer to create response.

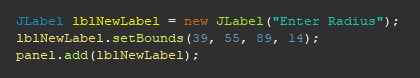
Event listeners: the are responsible for detecting or listening to the occurrence of the event and then response by executing the code, so when you press on the button the program will be executed.

Action performed: it is a part of the action listeners interface that is called when the event happen.

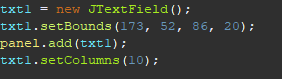
I used three ‘JButtons’ to allow user to calculate the area & volume of the shapes by clicking on them.



I used ‘JPanel’ of type ‘BevelBorder’ as a basic frame.

JLabel🡪 to display non-editable text.

Jtextfiel🡪 to accept input from user.

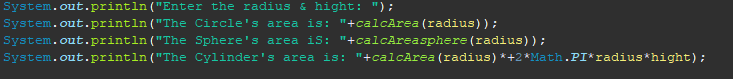


2.1.4:

Strengths in procedural paradigm code:

Modularity & organized: the code Is split into functions that encapsulate associated calculations, as each function do a different calculation, this improves code structure.

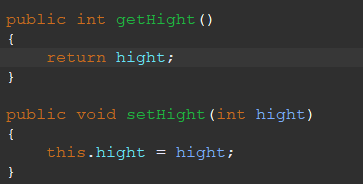
The code is clear, readable, and easy to understand as it has clear variables and function names indicating their use.

The included functions are used in multiple commands lines which applies code reusability. 

However might have code duplication the formula Math.PI\*Math.pow(radius, 2) is used in calcArea and calcAreasphere functions which makes it harder to update.

Object oriented strengths:

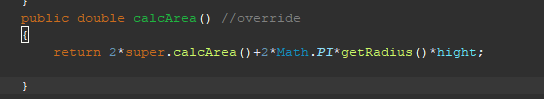
Encapsulation🡪 The circle and cylinder classes encapsulate height, radius and the calculation methods in wich they can be accessed through setters and getters.

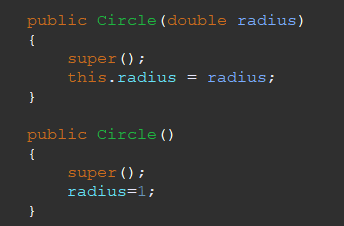


Inheritance🡪 The subclass cylinder inherits from the superclass circle the radius attribute.

Composition🡪 The inheritance connection allows the cylinder to extend from the circle class adding its own attribute.

Polymorphism🡪 the calcArea method overridden in the Cylinder class, there are constructors with different parameters which satisfies overloading.





However, the code could be improved, as the lack of interfaces and abstract classes in the code limits its flexibility and expansion.

Event driven:

While the code offers the basics for a shapes calculator, it may be enhanced in terms of GUI design, event handling, input validation, class design, and error handling.

Task2 of Part2:

2.2.1: firstly, I traced my code down and tried to analyze the code, where the error night has happened, then I used the debugging tools to step by step execute the code and track the error.

I faced different types of errors while writing the code such as:

1- Syntax errors: the most repeated syntax error is to forget writing a semicolon at the end of each line.

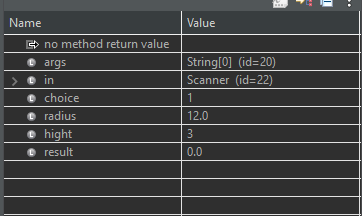
This type of error is easy to deal with as the compiler tells you where the error occurred.

2- Logical errors: logical errors happen when the results are wrong and isn’t as expected, this type of errors is the hardest to find, in my code I faced arithmetic logical error

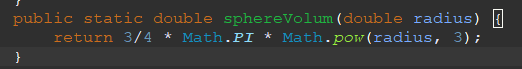
When calculating the sphere’s volume(3/4\*PI\*radius^3), I divided 3/4 as integers which resulted to zero.

I used the debugger tools to execute the code step by step using (step over), I also used (step into) to go through the method.









3- Run-time errors(exceptions): this type of error occurs during the execution of the code.

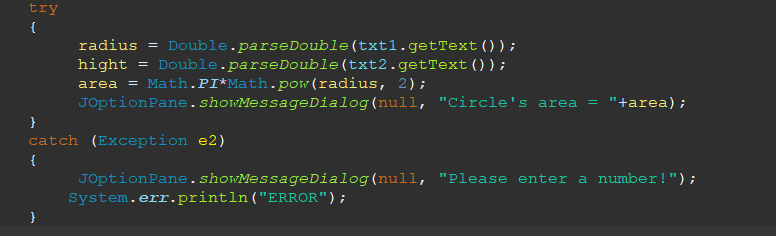
Exceptions are objects which are thrown at runtime.

Such as: array out of bounds, division by zero and input type mismatch.

In my code I used exception handling to maintain the code, as it might throw exceptions when specific inputs are entered.

Try/catch:

The try block contains the code that might have the exception. However, the catch block is where the execution codes, that will occur when the exception happens.



2.2.2:

Debugging: Debugging is the process of finding bugs and vulnerabilities in software code. programmers analyze the code to detect any errors that have happened.

Debugging process:

1. Error detection:

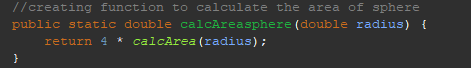
While debugging developers can test the code of the application to detect any possible flaws, they can identify where the weaknesses are by going through the code and test inputs, outputs and variables, which leads to improve the application and enhance its security.

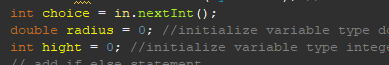
2- exception handling:

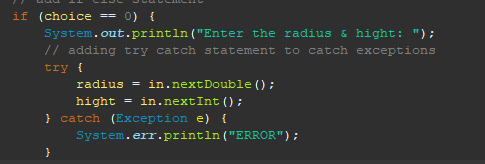
during the debugging process, may catch exception and errors while the program running, they may then apply error handling techniques to deal with such errors, avoiding possible security vulnerabilities like system crashing or information leakage.

2.2.3:

Coding standards:

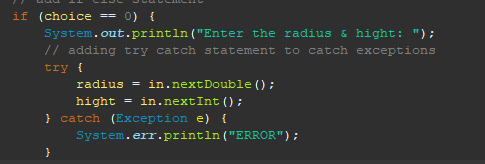
1- Naming conventions: I used camelCase to name methods, which starts with a lower-case letter, I also used PascalCase to name classes, which starts with an uppercase letter.

2- declarations 🡪 I used declaration per line to encourage commenting.

3- Indentation and formatting🡪 I used the recommended code indentation 

4- comments🡪 I used single line comments.

5- Try- catch statements🡪 I followed the recommended try-catch statement format.



Reference’s part 1:

1. <https://www.iste.org/explore/Computer-Science/Computer-programming-in-4-steps>
2. <https://www.geeksforgeeks.org/bubble-sort/>
3. <https://en.wikipedia.org/wiki/Bubble_sort>
4. <https://elearning.htu.edu.jo/pluginfile.php/120484/mod_resource/content/0/Algorithms.pdf>

Reference’s part 2:

1- <https://www.freecodecamp.org/news/what-exactly-is-a-programming-paradigm/>

