IBDP COMPUTER SCIENCE

Paper 2 [Option D]

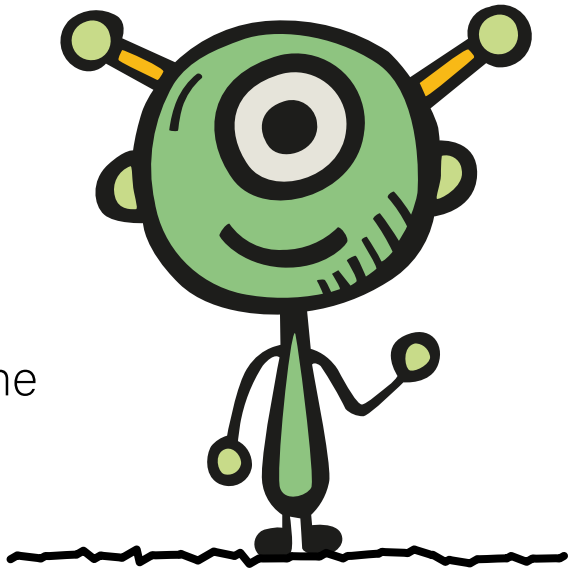
Time Limit: 1 hour

During this activity, you will not have access to technology, including a calculator. All you need is a pen or pencil and a problem-solving brain.

Instructions

If you are completing this activity with a partner/team, work together. Make sure you treat this activity as you would an exam paper - stay focussed and pay attention to the time limit.
The only difference is that you can share your ideas with a partner.

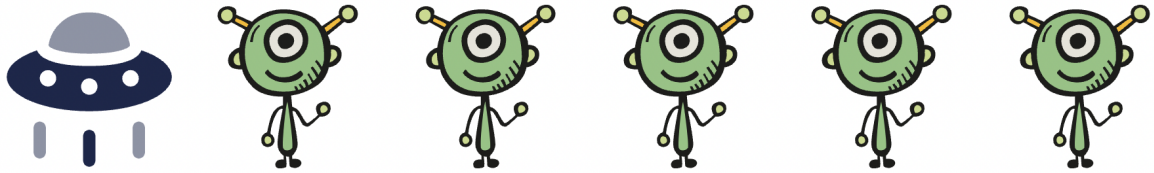
If you are completing this activity independently, make sure you treat this activity as you would an exam paper - stay focussed and pay attention to the time limit.



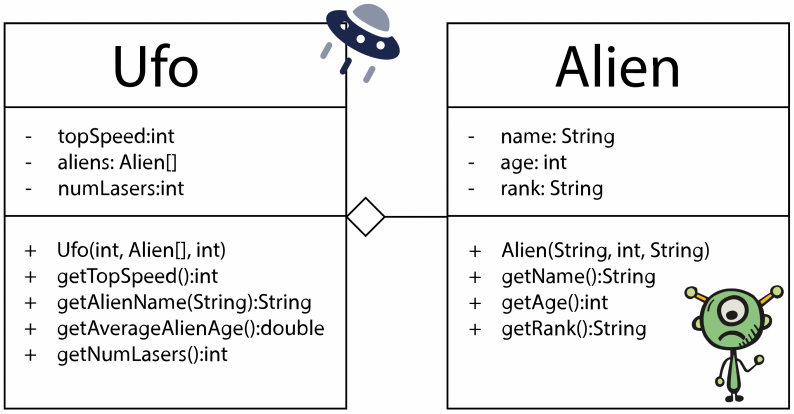
review activity

A Ufo has Aliens!

This is an example of containment/aggregation.



Here are UML diagrams describing the Ufo and Alien classes. *Note the position of the diamond - think of it as "has": the Ufo - HAS - Aliens*



Task 1

Design the full `Alien` class. Put a comment above each method noting if it a constructor, accessor or mutator method.

Also, as you are encapsulating properties and methods into the same class (a great strategy for managing data in a computer program), make sure you use the *data hiding* strategy.

You will need to use any of the following modifiers thoughtfully as you design properties and methods:

- public
- private
- protected
- static
- final

Note: the Constructor method takes a String,int, String as parameters. These represent *name,age,rank*.

```
// The Alien Class
```

Task 2

Design the full `Ufo` class.

Can you figure out how the Constructor method works? As with the `Alien` class, make sure you use data-hiding as you encapsulate properties and methods into the same class file. Consider also the modifiers you will use.

> The `getAlienName` method takes a String as a parameter. It represents a rank. It searches the `aliens` property and returns the name of any one Alien who has the same rank as the parameter.
>
> If no Alien in `aliens` has that rank, the method returns null.
>
> pre-condition: any of the elements in `aliens` may be null.
>
> post-condition: the method returns a String or null and the contents of `aliens` remains unchanged.

> The `getAverageAlienAge()` method calculates and returns the average age of every Alien in `aliens`.
>
> pre-condition: any of the elements in `aliens` may be null.
>
> post-condition: the method returns a double and the contents of `aliens` remains unchanged.

```
//The Ufo Class
```

Task 3

A new `Ufo` class method is required, `getOldestAlien()`.

This method sorts `aliens` in ascending order of age, and returns a reference to the object at the biggest index.

pre-condition: assume `aliens` is full of Alien object references.

post-condition: `aliens` is sorted by age and an Alien reference is returned.

Here is one attempted solution.

```
public Alien getOldestAlien(){

   for(int i = 0; i<aliens.length; i++){

      int smallest =

      for(int j = i+1; j<aliens.length; j++){

         if(aliens[j].        <aliens[        ].getAge() ){

            smallest = j;

         }

      }

      temp = aliens[i];

      aliens[i] = aliens[smallest];

      aliens[smallest] = temp;

   }

   return aliens[aliens.length-1];

}
```

Fill the blanks withe the correct code.

Task 4

The `Ufo` class requires another constructor.

This constructor will:

1. set the topSpeed to 0
2. initialise `aliens` as an array of 5 Alien objects
3. set the number of lasers to 3 (a standard number in the Alien community).

Add this constructor method to your design of the `Ufo` class.

State what OOP concept is being demonstrated here *hint: it is a form of POLYMORPHISM!*

Alternative Ufo class constructor method

Concept covered:

Task 5

Can you define the following terms? If there is +/- next to a term, can you also think of advantages and disadvantages?

- encapsulation +/-
- polymorphism
- aggregation
- constructor method
- modifier
- library routines/files +/-
- accessor method
- mutator method
- object reference
- parameter list
- method signature

DEFINITIONS