Examen 1

CI3641 - Lenguajes de programación I

Septiembre - Diciembre 2023

Amaranta Villegas

16-11247

Pregunta 1: Lenguaje escogido **Ada**

(a) De una breve descripción del lenguaje escogido.

i. Diga qué tipo de alcances y asociaciones posee, argumentando las ventajas y desventajas de la decisión tomada por los diseñadores del lenguaje, en el contexto de sus usuarios objetivos.

ii. Diga qué tipo de módulos ofrece (de tenerlos) y las diferentes formas de importar y exportar nombres.

iii. Diga si el lenguaje ofrece la posibilidad de crear aliases, sobrecarga y polimorfismo. En caso afirmativo, dé algunos ejemplos.

iv. Diga qué herramientas ofrece a potenciales desarrolladores, como: compiladores, intérpretes, debuggers, profilers, frameworks, etc.

Respuesta a.i

Ada al ser un lenguaje de programación que es calculado en tiempo de compilación posee alcance estático o léxico. Este lenguaje permite la detección de errores en sintaxis durante la fase de compilación y no durante la ejecución. Cuando se compila un programa Ada, el compilador analiza el código fuente y verifica que cumpla con las reglas sintácticas del lenguaje.

Este lenguaje también proporciona un mecanismo para manejar excepciones en tiempo de ejecución, de esta forma se pueden detectar y manejar errores que ocurren mientras el programa está en ejecución.

Es común que los lenguajes de programación con alcance estático también utilicen asociaciones profundas y Ada no es la excepción. En este lenguaje las variables internas de una subrutina toman el valor de una variable externa en el momento en el que se llama a la subrutina, es decir, que el valor de una variable es el que tiene en el momento en el que se hace la llamada a la subrutina y no el valor que pueda tener en un momento posterior, lo cual es característico de la asociación profunda.

La combinación de alcance estático y asociación profunda, permiten que los

programas en este lenguaje sean mucho más fáciles de depurar, más fáciles de leer y se ejecuten más rápido que los programas con alcance dinámico ya que el alcance de las variables no cambiará al momento de la ejecución del programa.

Respuesta a.ii

En Ada los módulos ofrecidos son conocidos como package. Los package en Ada se dividen en dos partes: especificación y cuerpo. La especificación del package es donde se define qué operaciones estarán disponibles para el uso de otros paquetes mientras que el cuerpo del package es donde se implementan las operaciones definidas en la especificación.

mostramos un ejemplo ilustrativos de las dos partes

especificación del package:

```
package Ejemplo is
    -- Declaraciones exportadas
    procedure Operacion_Exportada;
end Ejemplo;
```

La especificación del package siempre termina con la palabra reservada en seguida del nombre del package y un punto y coma, el cuerpo del package termina de la misma forma.

Cuerpo del package:

```
package body Ejemplo is
    procedure Operacion_Exportada is
    begin
        -- Código de la operación
    end Operacion_Exportada;
end Ejemplo;
```

Para importar un package en Ada, utilizamos la cláusula with y para exportar nombres desde un package se hace desde la especificación del package.
Todo subprograma, tipo o variable que esté declarado en la especificación podría

ser accesible desde cualquier lugar que se use with para importar el package.

Respueta a.iii

Para crear alias a una variable o a un tipo en Ada se utiliza la palabra reservada ***aliased***

un ejemplo para variables sería:

I : aliased Integer := 0;

para elementos de un array:

type Day_of Month is range 1 .. 31;
type Day_Has_Appintment is array (Day_of_Month) of aliased Boolean;

También se usa la palabra clave renames para crear alias a un componente de registro, lo cual es muy útil para simplificar la implementación de un subprograma.

Ejemplo:

```
Some Day : Date;
Y        : Integer renames Some_Day.Year;
```

En programación, se conoce como sobrecarga a la capacidad que tienen varios lenguajes de poder nombrar con un mismo identificador a diferentes variables u operaciones. En el caso de POO, cuando se habla de sobrecarga se refiere a la posibilidad de tener dos o más funciones con el mismo nombre pero con funcionalidades diferentes y es el compilador el que decide cuál de las funciones usará dependiendo de los parámetros que tenga cada quien.

En Ada diferentes subprogramas pueden compartir el mismo nombre por lo que sí existe la sobre carga, siempre y cuando las firmas del subprograma (nombre del subprograma, tipos de parámetros y tipos de valor devuelto) sean diferentes,

El compilador de Ada sabe que una asignación **a** requiere un archivo.

Por lo tanto, elige la función que devuelve **a** para satisfacer este requisito.

Esto se muestra en el siguiente ejemplo:

```
function Value (Str : String) return Integer;
function Value (Str : String) return Float;

V : Integer := Value ("8");
```

En el lenguaje de programación Ada, se puede lograr el polimorfismo a través de la programación basada en clases y un sistema de tipos de datos fuerte. Este lenguaje permite que dos objetos de distintas jerarquías de clases respondan a los mismos mensajes, a través de las denominadas interfaces.
Esto significa que dos objetos que implementen la misma interfaz podrán ser tratados de forma idéntica, como un mismo tipo de objeto, el tipo definido por la interfaz

Sin embargo, es importante mencionar que aunque Ada soporta el concepto de clases, no soporta completamente el polimorfismo, solo lo puede soportar a través de herencias y vinculaciones dinámicas.

Aquí tienes un ejemplo:

```
type Shape is tagged record
   X, Y : Float;
end record;

type Circle is new Shape with record
   Radius : Float;
end record;

procedure Draw (Obj : Shape'Class) is
begin
   -- código para dibujar la forma
end Draw;
```

En este ejemplo, Shape es una clase base y Circle es una clase derivada. El procedimiento Draw puede aceptar cualquier objeto de la clase Shape, incluyendo objetos de la clase Circle.

a.iv Herramientas que ofrece:

Para ***debugging*** tenemos Al Control que es una herramienta que verifica que el software Ada cumpla con las reglas definidas por el usuario y es de mucha utilidad para detectar errores de programación y asegurar la uniformidad del estilo de codificación.

Como ***framework*** para construir aplicaciones web y REST tenemos a AWS (Ada Web Server)

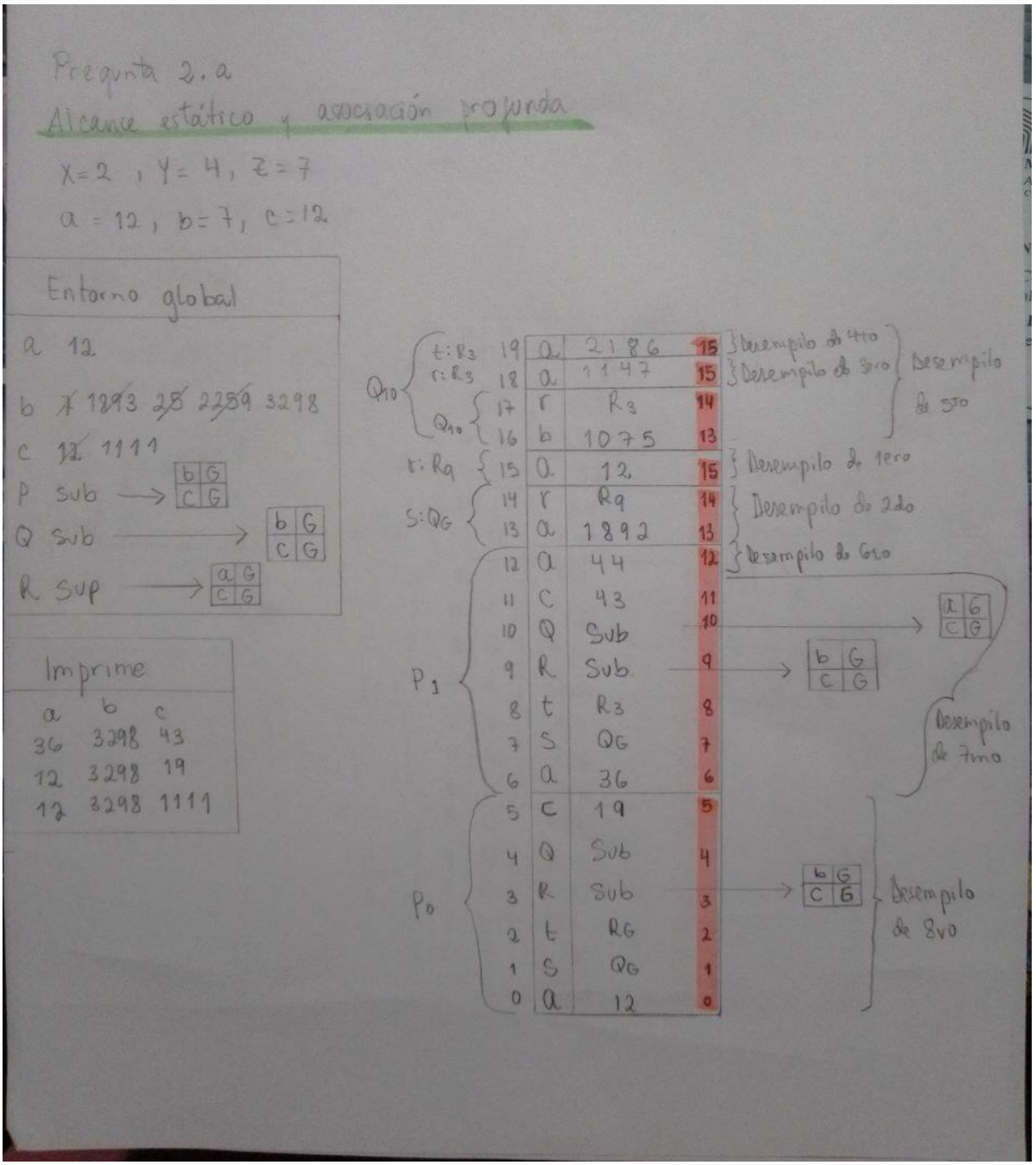
Como ***framework*** de pruebas unitarias para ADA está AdUnit.

Como ***IDE***, GNAT studio es un potente IDE para los desarrolladores de Ada, este soporta depuración, perfilado, funciones de autocompletado, arrastrar y soltar.

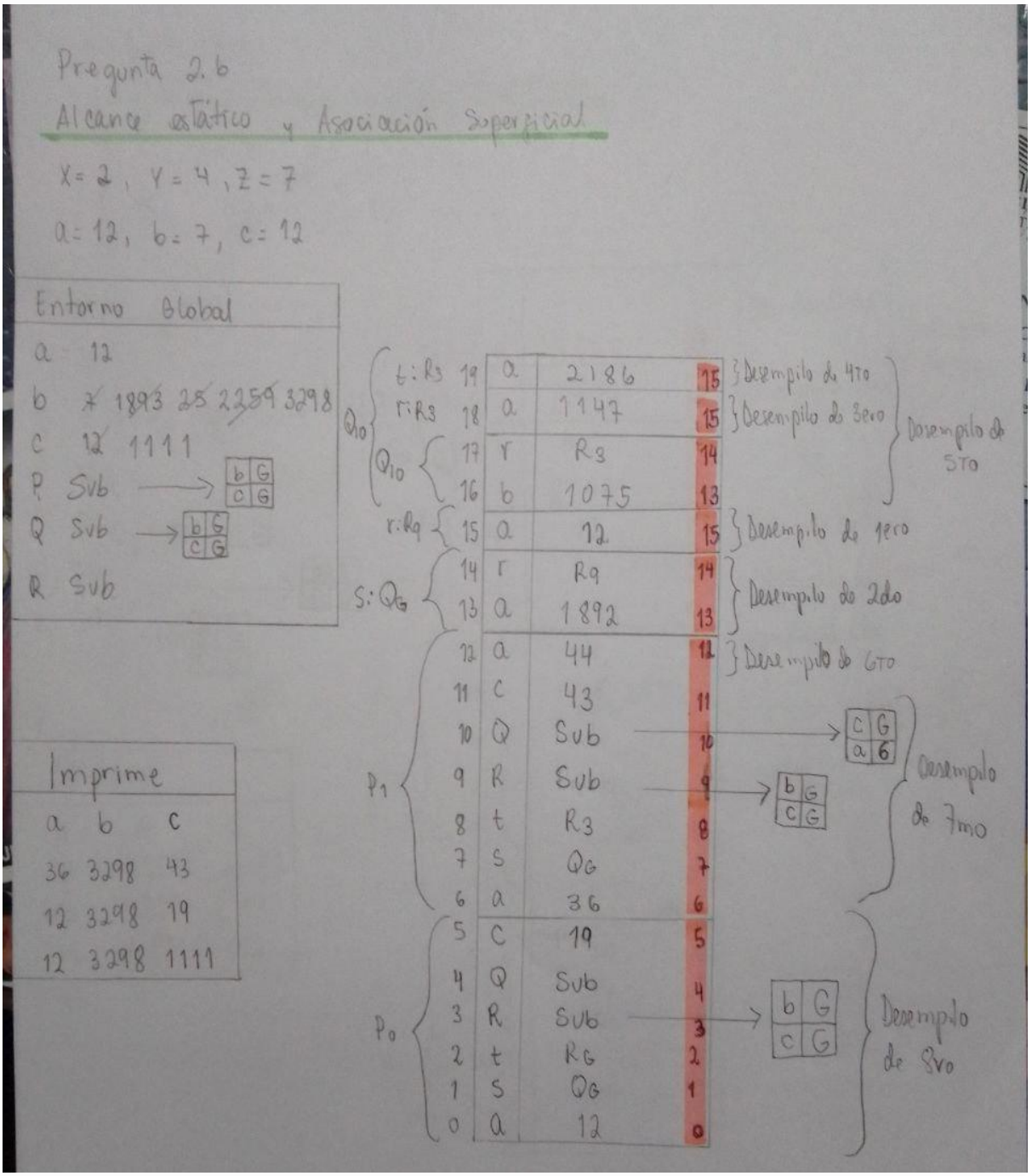
Pregunta 1.b puesta en el repositorio Examen1-Lenguajes1/pregunta1 at branch · amarantaVC/Examen1-Lenguajes1 (github.com)

Pregunta 2

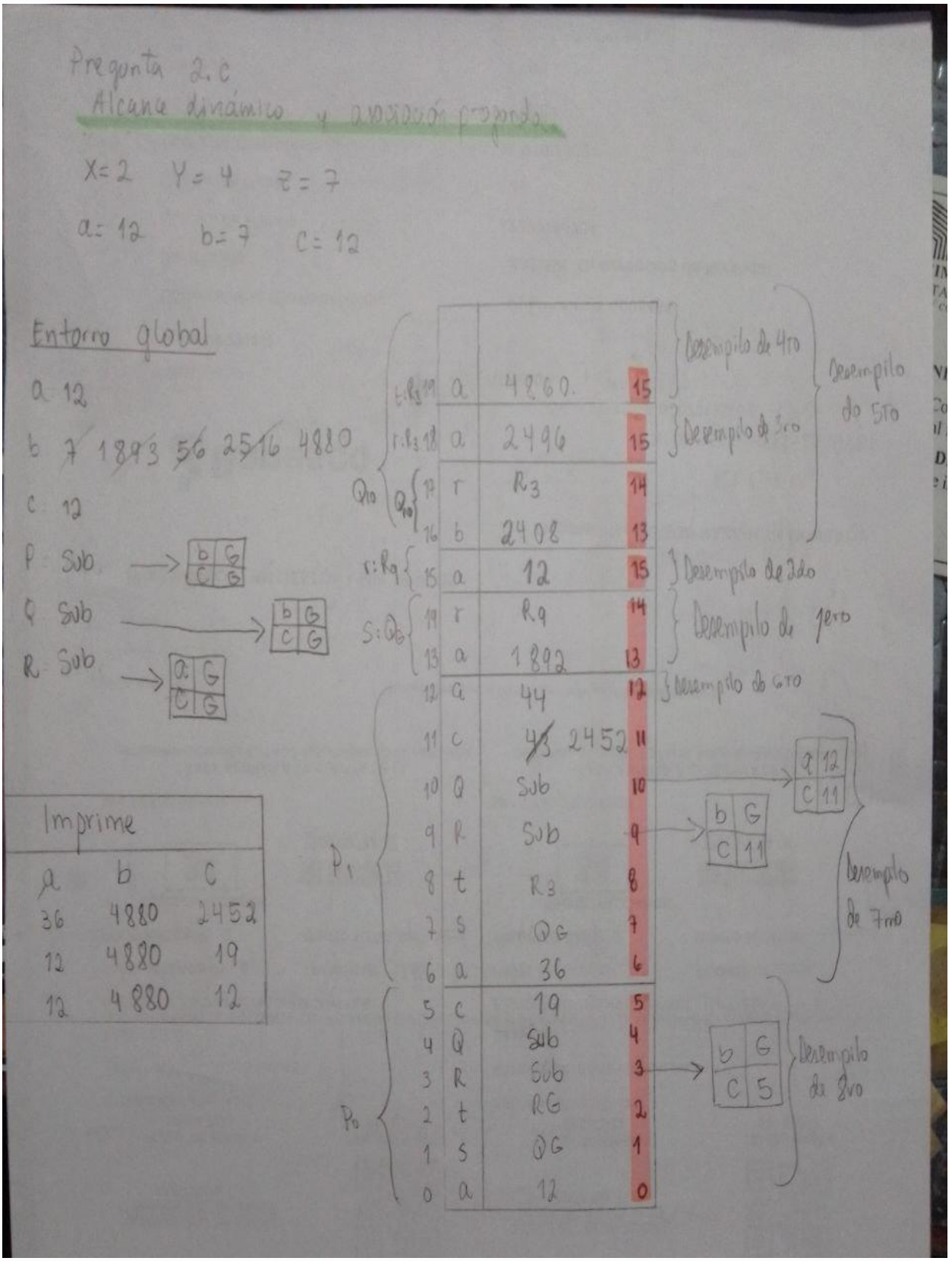
Alcance estático y asociacion profunda



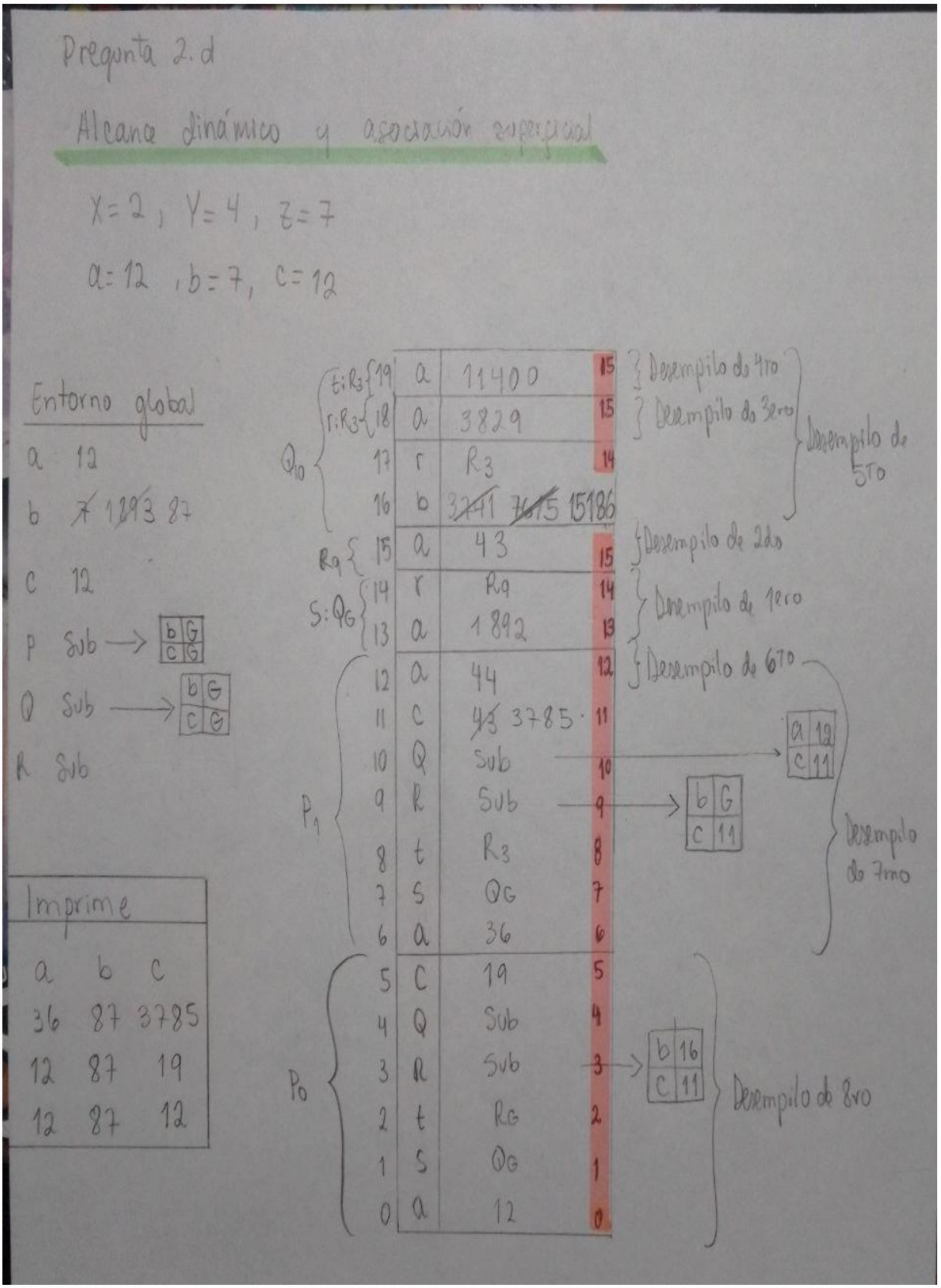
Alcance estático y asociación superficial



Alcance dinámico y asociación profunda



Alcance dinámico y asociación superficial



Pregunta 3. *Puesta en el repositorio*

Examen1-Lenguajes1/pregunta3 at branch · amarantaVC/Examen1-Lenguajes1 (github.com)

Pregunta 4. *Puesta en el repositorio*

Examen1-Lenguajes1/pregunta4 at branch · amarantaVC/Examen1-Lenguajes1 (github.com)