Prueba de síntesis 2021/22-2

Asignatura	Código	Fecha	Hora inicio
Prácticas de programación	75.555	18/6/2022	17:30

!!!	**Esta prueba solo pueden realizarla los estudiantes que hayan aprobado la evaluación continua**

Este enunciado también corresponde a las siguientes asignaturas:

- 81.578 - Prácticas de programación

Ficha técnica de la prueba

- No es necesario que escribas tu nombre. Una vez resuelta la prueba final, solo se aceptan documentos en formato .doc, .docx (Word) y .pdf.
- Comprueba que el código y el nombre de la asignatura corresponden a la asignatura de la que te has matriculado.
- Tiempo total: **1 hora** Valor de cada pregunta:
- ¿Puede consultarse algún material durante la prueba de síntesis? ¿Qué materiales están permitidos?
- ¿Puede utilizarse calculadora? ¿De qué tipo?
- Si hay preguntas tipo test, ¿descuentan las respuestas erróneas? ¿Cuánto?
- Indicaciones específicas para la realización de esta prueba de síntesis:

Prueba de síntesis 2021/22-2

Asignatura	Código	Fecha	Hora inicio
Prácticas de programación	75.555	18/6/2022	17:30

Enunciados

Pregunta 1 [50%]

Continuando la aplicación para la gestión de la pandemia de COVID, en este ejercicio se trabajará solo con los siguientes tipos:

```
const
    MAX_PERSONS: integer = 100;
    MAX_APPOINTMENTS: integer = 20;
end const

type
    tDate = record
        day : integer;
        month : integer;
        year : integer;
    end record

    tTime = record
        hour : integer;
        minutes : integer;
    end record

    tDateTime = record
        date : tDate;
        time: tTime;
    end record

    tVaccine = record
        name : string;
        required : integer;
        days : integer;
    end record

    tPerson = record
        document : string;
        name : string;
        surname : string;
        email : string;
        address : string;
        cp : string;
        birthday : tDate;
    end record

    tPopulation = record
        elems : vector[MAX_PERSONS] of tPerson;
        count : integer;
    end record
```

Prueba de síntesis 2021/22-2

Asignatura	**Código**	**Fecha**	**Hora inicio**
Prácticas de programación	75.555	18/6/2022	17:30

```
    tAppointment = record
        timestamp : tDateTime;
        person : tPerson;
        vaccine : tVaccine;
    end record

    tAppointmentData = record
        elems : vector[MAX_APPOINTMENTS] of tAppointment;
        count : integer;
    end record
end type
```

Dada la función:

> **function** population_with_vaccine(appointments: tAppointmentData, vaccine: string): tPopulation

que dadas unas citas y una vacuna devuelve la lista de personas que tienen una cita programada para esa vacuna. Para simplificar el ejercicio podemos asumir que la misma persona tendrá como máximo una cita, es decir, una persona no podrá tener dos o más citas programadas.

Se pide:

a) **[10%]** Indica las pre-condiciones para la función *population_with_vaccine*. En este caso, se asume que la lista de citas no está vacía y que existe por lo menos alguna persona con cita para la vacuna que nos dan. **Justifica la respuesta con una o dos líneas de texto**.

b) **[40%]** Implementa la función en **lenguaje algorítmico** aplicando la metodología de diseño descendente. También es necesario implementar todas las funciones que necesites crear en los diferentes niveles. Para cada método, describe brevemente la función que realizan (escribe una o dos líneas de texto por método). **No se valorarán métodos que no tengan su correspondiente descripción**.

Prueba de síntesis 2021/22-2

Asignatura	**Código**	**Fecha**	**Hora inicio**
Prácticas de programación	75.555	18/6/2022	17:30

Pregunta 2 [50%]

Dada la siguiente implementación en lenguaje C de un método *mystery*, donde la función:
int search(char a, char c[N])
busca el carácter *a* en la cadena *c*, retornando 1 si lo encuentra, 0 en caso contrario:

```
#define N 128
int mystery (char p[N], int x, char q[N])
{
        assert (-1 <= x && x < N);
        int count;
        if (x < 0)                                              (1)
                count = 0;                                      (2)
        else                                                    (3)
                count = mystery (p, x-1, q) + search(p[x], q);  (4)
        return count;                                           (5)
}
```

a) [10%] Explica qué hace este método *mystery*, es decir, qué función realiza. También muestra paso a paso (incluyendo un diagrama de cajas para secuencia de llamadas y sus parámetros) el funcionamiento de *mystery* para la siguiente invocación:

```
int ret;
ret = mystery("exam", 3, "hexagon" );
```

b) [5%] Explica qué consecuencias tendría sustituir la línea (4) por esta línea:

```
count = mystery (p, x, q) + search(p[x], q);
```

c) [20%] Transforma este método recursivo en un método iterativo en lenguaje C. **Describe brevemente la implementación adoptada** (en caso contrario, no se evaluará el ejercicio).

d) [15%] Analiza la complejidad de este método (en su versión recursiva) asumiendo que la función de tiempo de search(a, c) es T_{search} = s, donde s es la longitud de c en carácteres.