Universidad de Santiago de Chile
Facultad de Ingeniería
Departamento de Ingeniería Informática
Paradigmas de Programación

Paradigmas de Programación
Proyecto 2 - Programación Funcional

Enunciado

Un nuevo centro clínico ha sido abierto en el centro de Santiago llamado *FastClinic* y requiere un sistema computacional para administrar las fichas de los pacientes las cuales actualmente se encuentran registradas en una base datos heredada de un antiguo sistema informático para el cual ya no se cuenta con soporte. El funcionamiento básico de este centro clínico es así: Los pacientes llegan al centro clínico por un motivo de consulta, como por ejemplo, dolor de estómago, tos, dolor de cabeza, etc. Los médicos examinan a los pacientes y obtienen un diagnóstico. Dependiendo del diagnóstico dado, es el tratamiento que se le da al paciente.
La clínica además nos ha dado la siguiente información respecto de los datos con los que cuentan y las funcionalidades que se necesitan implementar en el nuevo sistema:

1. Un paciente es atendido por uno o más médicos y un médico puede atender múltiples pacientes.
2. Un paciente puede llegar con uno o varios motivos de consultas.
3. Un paciente puede tener uno o varios diagnósticos.
4. Los diagnósticos son efectuados por un médico responsable en una fecha (dd/mm/aaaa) particular.
5. Los pacientes pueden recibir distintos tratamientos recomendados por un doctor. Los tratamientos están normalizados para cada diagnóstico.
6. Los diagnósticos pueden ser tratados con uno o más tratamientos. Los tratamientos pueden ser comunes para uno o más diagnósticos.
7. Los pacientes reciben el alta de un diagnóstico en una fecha (dd/mm/aaaa) particular y bajo la responsabilidad de un médico (que puede ser diferente del médico que hizo el diagnóstico original).

Respecto de la base de datos existente se tiene conocimiento de las tablas y sus campos:

1. Paciente: IdentificadorPaciente, rut, email, nombre, apellido, fecha de nacimiento (dd/mm/aaaa)
2. Médico: IdentificadorMedico, rut, email, nombre, apellido, especialidad.
3. Diagnóstico: identificadorDiagnostico, nombre del diagnóstico, nivelGravedad (nulo, bajo, medio, alto, muy alto)
4. Tratamiento: identificadorTratamiento, nombre, descripción, nivel de riesgo (nulo, bajo, medio, alto, muy alto).

5. TratamientoDiagnostico: identificadorDiagnostico, identificadorTratamiento
6. Vinculación TratamientosDiagnosticoPaciente: identificadorDiagnosticoPaciente, identificadorTratamiento, identificadorMedico, fechaInicio (dd/mm/aaaa), duracion y resultado (ej. exitoso, parcial, fracaso)
7. Vinculación DiagnosticoPaciente: identificadorDiagnósticoPaciente, identificadorPaciente, identificadorDiagnóstico, fechaDiagnóstico, identificadorMedicoDiagnostico, estadoDiagnostico, fechaAlta (dd/mm/aaaa), identificador MedicoAlta y detalleAlta.

Desarrollar un programa en el lenguaje de programación Scheme que permita a través de interfaz de línea de comandos realizar operaciones de manipulación y consulta - especificadas como parámetros sobre un conjunto de datos de entrada contenidos en listas (archivos de texto plano).

Como parte del diseño e implementación de su solución se requiere que implemente una representación para cada una de las entidades (tablas) y sus relaciones. Las representaciones internas que decida emplear como parte de su TDA (Tipo de datos abstractos) son una decisión personal. Además de la representación, debe crear constructores apropiados, funciones de pertenencia, selectores necesarios para extraer información de su representación, y modificadores necesarios.

Considere la siguiente estructura de TDA:

Funciones que operan sobre el nuevo tipo de dato
Modificadores
Selectores
Funciones de pertenencia
Constructores
Representación

- **Los constructores básicos solicitados son:**

A. (createPaciente ID rut email nombre apellido fechaNacimiento)
Ej: (createPaciente 14 “17123123-5” “algo@usach.cl” “juan” “perez”, “20/08/1945”)

B. (createDoctor ID rut email nombre apellido especialidad)
Ej: (createDoctor 2 “6522511-6” “eldoc@usach.cl” “David Ernesto” “King”)

C. (createDiagnostico ID descripcionDiagnostico nivelGravedad)
Ej: (createDiagnostico 4 “tuberculosis” “alto”)

D. (createTratamiento ID nombreTratamiento descripcionTratamiento nivelDeRiesgo)
Ej: (createTratamiento 5 “reposo” “cama durante una semana” “bajo”)

E. (createDiagnosticoPaciente idPaciente idDiagnostico fechaDiagnostico idDoctorDiagnostico estadoDiagnostico fechaAlta idDoctorAlta detalleAlta)
Ej: (createDiagnosticoPaciente 14 4 “12/12/2000” 2 “vigente” “05/01/2001” 14 “necesita silla de ruedas para salir del hospital”)

F. (createTratamientoDiagnostico idDiagnostico idTratamiento)
Ej: (createTratamientoDiagnostico 4 5)

G. (createTratamientoDiagnosticoPaciente ID idTratamiento idDoctor fechaInicio duracionDias resultado)
Ej: (createTratamientoDiagnosticoPaciente 13 5 2 “01/01/2001” 4 “exitoso”)

Note que pueden haber espacios en algún campo, pero para que no sea considerado otro argumento se utilizan comillas para el argumento. Los constructores corresponden a funciones en scheme, los cuales deben utilizarse por las consultas que se indican más adelante.

- **Las funciones de pertenencia requeridas son (0.1 cada una):**

Debe implementar al menos 4 funciones de pertenencia, si realiza más implicarán 0.1 puntos por cada una extra realizada.

A. (isPaciente? [supuestoPaciente])
Ej: (isPaciente? (createPaciente 14 “17123123-5” “algo@usach.cl” “juan” “perez” “20/08/1945”))
Salida: #t

B. (isDoctor? [supuestoDoctor])
Ej: (isDoctor? (createDoctor 2 “6522511-6” “eldoc@usach.clq” “David Ernesto” “King”))
Salida: #t

C. (isDiagnostico? [supuestoDiagnostico])
Ej: (isDiagnostico? (createDiagnostico 4 “tuberculosis” “alto”))
Salida: #t

D. (isTratamiento? [supuestoTratamiento])
Ej: (isTratamiento? (createTratamiento 5 “reposo” “cama durante una semana” “bajo”))
Salida: #t

E. (isDiagnosticoPaciente? [supuestoDiagnosticoPaciente])
Ej: (isDiagnosticoPaciente? (createDiagnosticoPaciente 14 4 “12/12/2000” 2 “vigente” “05/01/2001” 14 “necesita silla de ruedas para salir del hospital”))
Salida: #t

F. (isTratamientoDiagnostico? [supuestoTratamientoDiagnostico])
Ej: (isTratamientoDiagnostico? (createTratamientoDiagnostico 4 5))
Salida: #t

G. (isTratamientoDiagnosticoPaciente? [supuestoTratamientoDiagnosticoPaciente])
Ej: (isTratamientoDiagnosticoPaciente? (createTratamientoDiagnosticoPaciente 13 5 2 “01/01/2001” 4 “exitoso”))
Salida: #t

Todas estas funciones de pertenencia deben devolver #t o #f dependiendo si su único argumento es realmente del tipo de dato que se consulta o no.

- Las funciones de selección requeridas son:

Nivel bajo (0.2 pts cada una):

A. Obtener el nombre de un paciente dado su rut.
Formato: (obtenerNombrePaciente [rut])
Ej: (obtenerPaciente “12345789-0”)
Salida: “Alvaro”

B. Obtener la especialidad de un médico a partir de su rut.
Formato: (especialidad [rut])
Ej: (especialidad “2222-2”)
Salida: “Dermatologia”

C. Listar el (o los) tratamiento(s) de acuerdo a un nivel de riesgo dado, deben mostrarse en el mismo orden en que aparecen en la tabla “Tratamiento”.
Formato: (tratamientoRiesgoso [nivelRiesgo])
Ej: (tratamientoRiesgoso “alto”)

Salida: “Radioterapia”
“Quimioterapia”
“Cirugía”

D. Determinar cuántos médicos han indicado un determinado tratamiento.

Formato: (cantidadMedicosTratamiento [idTratamiento])
Ej: (cantidadMedicosTratamiento 3)
Salida: 10

E. Determinar el identificador de los médicos que han realizado un diagnóstico determinado, estos deben mostrarse en orden creciente.

Formato: (medicoDiagnostico [idDiagnostico])
Ej: (medicoDiagnostico 555)
Salida: 8
12
18

F. Cuántos pacientes ha tratado un médico a partir de su id.

Formato: (cantidadPacientesMedico [idMedico])
Ej: (cantidadPacientesMedico 44)
Salida: 40

G. Determinar el porcentaje de éxito de un tratamiento a partir de su id.

Formato: (exitoTratamiento [idTratamiento])
Ej: (exitoTratamiento 8765)
Salida: “35%”

Nivel medio (0.4 pts cada una):

H. Determinar el (o los) tratamiento(s) más usado(s) para para un diagnóstico particular indicando la cantidad de veces que se ha empleado, deben mostrarse en el mismo orden que la tabla “Tratamiento”.

Formato: (tratamientoMasUsado [idDiagnostico])
Ej: (tratamientoMasUsado 1234)
Salida: “Radioterapia 5”
“Quimioterapia 5”

I. Identificar el (o los) médico(s) que más altas otorga indicando su rut, nombre y apellido, y la cantidad de altas, el rut debe separarse mediante una coma del nombre y el apellido, lo mismo debe ocurrir para la cantidad de altas. En caso de haber dos o más médicos con igual cantidad de altas se deben mostrar todos estos ordenados según como aparecen en la tabla “Doctor”.

Formato: (medicoMasAltas)

Ej: (medicoMasAltas)
Salida: “15156-5,Juan Perez,9”

J. Identificar el (o los) tratamiento(s) más usado(s) en todo el sistema indicando la cantidad de veces que se ha(n) usado, En caso de haber dos o más tratamientos con igual cantidad de veces que han sido usados, se deben mostrar todos estos ordenados según como aparecen en la tabla “Tratamiento”.

Formato: (tratamientoMasUsado)
Ej: (tratamientoMasUsado)
Salida: “dieta balanceada,8”

K. Dado un rut del paciente, determinar su(s) diagnóstico(s) más frecuente(s).

Formato: (rutPacienteDiagnostico [rutPaciente])
Ej: (rutPacienteDiagnostico 6767-4)
Salida: “Apendicitis”

L. Determinar qué tratamientos puede recibir un paciente a partir de su diagnóstico. Se debe mostrar el nombre del tratamiento concatenado a un espacio y concatenado a la descripción del tratamiento, estos deben mostrarse en el mismo orden en que aparecen en la tabla “Tratamiento”.

Formato: (tratamientosDiagnosticoPaciente [idDiagnostico] [identificadorPaciente])
Ej: (tratamientosDiagnosticoPaciente 7654 12343)
Salida: “Reposo en cama por 3 días”
“Paracetamol 500 mg cada 8 horas”

M. Dado el identificador de un paciente, indicar el rut, nombre y apellido de todos los médicos que lo han tratado (no incluir los que lo dieron de alta), estos deben mostrarse en el mismo orden en que aparecen en la tabla “Doctor”.

Formato: (medicosTratantes [identificadorPaciente])
Ej: (medicosTratantes 67)
Salida: “1335552-3 Juan Gonzalez”
“5346222-2 Pedro Quintana”
“5833444-9 Felipe Acevedo”

N. Listar los tratamientos que ha recibido un paciente a partir de su correo electrónico. Se debe mostrar el nombre del tratamiento concatenado a un espacio y concatenado a la descripción del tratamiento, estos deben mostrarse en el mismo orden en que aparecen en la tabla “Tratamiento”.

Formato: (tratamientosPacienteCorreo [correoElectronico])
Ej: (tratamientosPacienteCorreo “miCorreo@servidor.com”)
Salida: “Reposo en cama por 3 días”
“Paracetamol 500 mg cada 8 horas”
“Radioterapia Quimioterapia”

Nivel alto (1 pts cada una):

O. Dado el nombre y apellido de una persona, indicar cuál es el nivel de riesgo del último tratamiento recibido.

Formato: (riesgoUltimoTratamiento [nombre] [apellido])
Ej: (riesgoUltimoTratamiento “Juan” “Perez”)
Salida: “alto”

P. Conocer todos los pacientes diagnosticados con el diagnóstico X (nombre del diagnóstico) a los cuales el doctor Y (rut) les dio el Alta. Debe mostrar el rut, nombre y apellido. Estos deben mostrarse en el mismo orden en que aparecen en la tabla “Paciente”, no deben repetirse resultados.

Formato: (diagnosticoPacienteMedico [nombreDiagnostico] [rutDoctor])
Ej: (diagnosticoPacienteMedico “Resfrio” “1112-3”)
Salida: “67643-9,Carlos,Ibanez”
“87538-k,Isaias,Dominguez”

Q. Dado el rut de un paciente, indicar el rut, nombre y apellido de todos los médicos que lo han tratado (no incluye a los que lo han dado de alta), estos deben mostrarse en el mismo orden en que aparecen en la tabla “Doctor”.

Formato: (listarMedicosTratantesPaciente [rutPaciente])
Ej: (listarMedicosTratantesPaciente “78654-8”)
Salida: “45454-4,Pedro,Montoya”
“3233232-1,Rocio,Jimenez”
“77777-8,Esteban,Lazaro”

Las funciones de modificación requeridas son:

Además se requiere hacer algunas modificaciones a los datos, estas se realizan en directamente en los archivos de entrada sin pasar por la salida por pantalla.

Nivel bajo (0.2 pts cada una):

R. Modificar el correo electrónico de un paciente. Se asume que el correo es único.

Formato: (modificarCorreoPaciente [correoAntiguo] [correoNuevo])
Ej: (modificarCorreoPaciente “miCorreo1@server.com” “miCorreo2@server.com”)

S. Modificar el resultado de un tratamiento de un paciente.

Formato: (modificarEstadoPaciente [idPaciente] [idTratamiento] [resultado])
Ej: (modificarEstadoPaciente 1 2 “exito”)

T. Cambiar la descripción de un tratamiento a partir de su id.

Formato: (modificarDescripcionTratamiento [idTratamiento] [descripcion])
Ej: (modificarDescripcionTratamiento 12 “Reposo por 5 días”)

Nivel medio (0.4 pts cada una):

U. Eliminar un paciente, siempre y cuando este no tenga diagnósticos.

Formato: (eliminarPaciente [rutPaciente])
Ej: (eliminarPaciente “99898-6”)

V. Eliminar un médico, siempre y cuando este no sea responsable de diagnósticos o tratamientos vigentes.

Formato: (eliminarMedico [rutMedico])
Ej: (eliminarMedico “67643-9”)

W. Dado el nombre y apellido de un paciente, además del id de un tratamiento, modificar el resultado de su tratamiento más reciente (puede darse en caso que un paciente reciba varias veces el mismo tratamiento).

Formato: (modificarResultadoTratamiento [nombre] [apellido] [idTratamiento] [nuevoResultado])
Ej: (modificarResultadoTratamiento “Juan” “Soto” 123 “se aprecia recuperación”)

Nivel alto (1 pts cada una):

X. Eliminar un tratamiento siempre y cuando no esté vinculado a tratamientos en curso. La eliminación de un tratamiento también involucra la eliminación de este de su vinculación con el o los diagnósticos.

Formato: (eliminarTratamiento [idTratamiento])
Ej: (eliminarTratamiento 1234)

Y. Dado el correo del paciente y del médico, modificar el médico que da el alta en su última ocasión.

Formato: (modificarMedicoAlta [correoPaciente] [correoMedico])
Ej: (modificarMedicoAlta “correoPaciente@server.com” “correoMedico@server.com”)

Evaluación:

La evaluación del proyecto considera una serie de requerimientos mínimos para poder optar por una calificación 4.0 en ejecución; se deben implementar todos los constructores, 4 funciones de pertenencia, 4 requerimientos de nivel bajo, 2 de nivel medio y 2 de nivel alto. Debe de haber por lo menos, una de consultas y una de modificación en cada nivel.

Para optar a una nota superior, deberá sumar puntaje implementando otras funciones del listado propuesto anteriormente. Sin embargo, debe considerar que en total no puede exceder 6 requerimientos de nivel bajo y 6 de nivel medio. Para las funciones de pertenencia y los de nivel alto no hay límites. La nota máxima a la que puede aspirar es un 7.0.

Respecto de la evaluación final del proyecto:

Si (NotaImplementación >=4.0)
    NotaFinal = NotaImplementación*0.8+NotaInforme*0.2
Sino
    NotaFinal = NotaImplementación

Indicaciones Generales:

Debe utilizar DrRacket para ejecutar su código utilizando la directiva "#lang racket" y todas las funciones que puedan ser llamadas externamente (los constructores, funciones de pertenencia, funciones de selección y de modificación) deben declararse utilizando la función "provide" de racket, de esta forma su archivo principal de código quedará como se muestra a continuación:

```
#lang racket
;comentario de ejemplo: Acá comienza la declaración de las funciones que pueden ser
; llamadas externamente, se deben poner todas las funciones que son indicados
; en los requerimientos y que usted haya implementado utilizando la función 'provide'.
(provide funcionImplementada1)
(provide funcionImplementada2)
(provide funcionImplementada3)
…
…
; Otro comentario: acá comienza la definición de las funciones anteriormente declaradas
(define (funcionImplementada1 param1 param2 etc) (cuerpoFuncionImplementada1)
(define (funcionImplementada2 param1 param2 etc) (cuerpoFuncionImplementada2)
(define (funcionImplementada3 param1 param2 etc) (cuerpoFuncionImplementada3)
…
…
```

Para probar su código desde racket, basta llamar la función correspondiente de la misma forma en que se mostró en los ejemplos de cada requerimiento a implementar. Sin embargo, el código también puede ser ejecutado desde la consola (en GNU/Linux) sin abrir el IDE de racket utilizando el siguiente comando:

```
racket -e '(require "fastClinic.rkt")(funcionALlamar param1 param2 etc)'
```

Notar que para que el comando funcione, el archivo de código que tiene las funciones que pueden ser llamadas debe llamarse “fastClinic.rkt” y estar en el mismo directorio en que se ejecuta el comando por consola.

El alumno debe hacer explícito qué funcionalidades fueron implementadas y están en funcionamiento para así no afectar el proceso de revisión. Si una función en particular no ejecuta o es inestable, dejarla comentada en el código.
Debe incluir informe técnico de su desarrollo. Los detalles del formato de este documento se encuentran en el documento en el aula virtual del curso.

1) Realice una descomposición de funciones adecuada.
2) Debe demostrar el uso de recursión de cola y recursión lineal. Al menos una función de las implementadas debe ser implementada con recursión de cola o al menos una función debe ser implementada con recursión lineal. En el resto de las funciones usted decide qué tipo de recursión usar. Mediante comentarios debe indicar el tipo de recursión empleado.
3) En cada función señalar como comentario su firma (signature), dominio de sus argumentos y su recorrido (que retorna).
4) Documentar el código indicando el propósito de las funciones y sus argumentos (además del signature señalado anteriormente).
5) Cabe mencionar que en cada uno de los archivos de entrada los datos en cada registro están separados por comas, se hará uso de los mismos archivos de entrada del laboratorio 1.
6) Las salidas deben realizarse utilizando salida estándar y deben terminar con un salto de línea (\n). De tratarse de múltiples registros o datos de salida, cada registro debe aparecer en una nueva línea. No se debe desplegar ninguna otra información por pantalla.
7) Debe respetar totalmente el formato especificado para los llamados al programa (nombre de parámetros, orden, uso de mayúsculas y minúsculas). No se hacen nuevas revisiones/evaluaciones por fallas en seguir las normas de formato.
8) Los argumentos del programa deben ser como aparece. Lo que se ingrese, busque, elimine, etc. son ejemplos, por lo que pueden ser cambiados al evaluar su programa.
9) Parámetros de entrada al momento de llamar una función cuyo valor sea texto con espacios, se deben usar comillas (ej. “Esto es un parámetro”).
10) Cualquier falta en relación al formato, archivos de origen, etc. Será evaluado con la nota mínima.
11) El alumno debe hacer explícito en el informe qué funcionalidades fueron implementadas y están en funcionamiento para así no afectar el proceso de revisión. Si una función en particular no ejecuta o es inestable, dejarla comentada en el código.
12) Se debe entregar un informe con las siguientes secciones: Introducción, Descripción del Problema, Objetivos, Solución, Resultados, Conclusión, Instrucciones de uso, Referencias, Anexo Código fuente. Procure usar fuente Times New Roman, tamaño 12, espaciado 1.5. Máximo 15 planas de contenido (se excluyen de este número de páginas

portada, tabla de contenidos, y apéndice con código fuente cuyo formato de entrega está especificado en un documento que se encuentra disponible en UsachVirtual.

13) Además del informe se debe entregar código fuente documentado e impreso por ambos lados, usar tamaño de fuente adecuado para que lineas de código se ajusten al tamaño de la hoja (sólo para el código fuente pueden reducir el tamaño de fuente o cambiarlo por uno más adecuado). Esté se debe entregar en horario de clases al profesor de la coordinación correspondiente.
14) La entrega será el día viernes 17 de octubre a las 23:55 por usachvirtual, la clase siguiente a ese plazo debe entregar el material impreso que se indicó anteriormente.
15) Cambios en el enunciado se especificarán en UsachVirtual.

Cantidad de requerimientos totales

(Esta hoja es sólo una forma de resumir lo pedido)

	Bajo	Medio	Alto	Total
Constructores				7 (min 7)
Funciones de pertenencia				7 (min 4)
Funciones de selección	7 (min 1)	7 (min 1)	3 (min 1)	
Funciones de modificación	3 (min 1)	3 (min 1)	2 (min 1)	
Se piden en la columna:	min 4; max 6	min 2; max 6	min 2	min 7 y min 4