Práctica 2

Llamada a procedimiento remoto (RPC)

Parte 1 - RPC Sun

Luis Miguel Guirado Bautista
Curso 2024-2025
Universidad de Granada

Introducción

Este documento contiene información acerca del desarrollo de la calculadora que se ha pedido para esta práctica. Se hablará sobre la definición y representación de datos compartidos, el funcionamiento del programa cliente y servidor, la comunicación entre estos y las salidas que muestran

Representación de datos compartidos

Los datos compartidos se definen en el fichero 'calculadora.x', así como la definición del programa y sus versiones. Se han definido los datos necesarios para la implementación de operaciones basicas, vectoriales de hasta 10 componentes y operaciones compuestas

Parte troncal. Operaciones básicas

Las operaciones básicas son las operaciones con dos operandos y un operador
Se han definido dos tipos

- `Numero` como `double`
- `Operador` como `char`

También se ha definido una estructura para almacenar los datos de una operacion

- Dos componentes tipo Numero para guardar los operandos
- Un componente tipo Operador para guardar el operador

Finalmente se ha definido una unión llamada ResultadoNumerico que, si el código de error de la unión es 0, guardará un Numero que contendrá el resultado de la operación.

```
/* Definicion de tipo para los operandos o datos numericos */
typedef double Numero;

/* Definicion de tipo para los operadores */
typedef char Operador;

/* Definicion de estructura para definir operaciones simples */
struct Operacion {
        Numero operando1;
        Operador operador;
        Numero operando2;
};

/* Definicion de resultado de una operacion basica */
union ResultadoNumerico switch (int errnum) {
        case 0:
             Numero resultado;
        default:
             void;
};
```

Parte adicional

Operaciones con vectores

Las operaciones con vectores son las operaciones con dos vectores, que contienen un número de valores igual entre ellos y un operador (exceptuando el operador de división)
Se ha definido

- Una constante MAX_VECTOR para limitar el tamaño de los vectores igual a 10
- Un tipo ValoresVector como un array de tipo Numero de como máximo MAX_VECTOR elementos
- Una estructura llamada Vector que contiene la dimension del vector como entero y un miembro ValoresVector para guardar los valores (se concluye que esto resulta redundante ya que rpcgen genera para las definiciones de arrays un struct que guarda los valores y el tamaño del mismo)
- Una estructura llamada OperacionVectorial que guarda dos Vectores y un Operando
- Una union llamada ResultadoVectorial para guardar el resultado de una operacion vectorial. Es igual a ResultadoNumerico pero que en vez que guardar un Numero guarda un Vector

```
/* Definicion de valores del vector */
const MAX_VECTOR = 10;
typedef Numero ValoresVector<MAX_VECTOR>;

/* Definicion del vector, especificamos dimension y valores */
struct Vector {
        int dim;
        ValoresVector valores;
};

/* Definicion de operacion vectorial */
struct OperacionVectorial {
        Vector vector1;
        Operador operador;
        Vector vector2;
};

/* Definicion de resultado de una operacion vectorial */
union ResultadoVectorial switch (int errnum) {
        case 0:
             Vector resultado;
        default:
             void;
};
```

Operaciones compuestas

Las operaciones compuestas son aquellas operaciones numericas que contienen al menos 3 operandos y 2 operadores y que el número de operadores es igual al de operandos menos 1. Para este tipo de operaciones se ha tenido en cuenta una prioridad de operadores definida por el programador segun el orden de aparición (siendo la izquierda el lado más prioritario) y el tipo de operador (primero las multiplicaciones, luego las divisiones y finalmente las sumas y las restas)
Se ha definido una estructura con un array dinámico de operandos tipo Numero y un array dinámico de operadores tipo Operador

```
/* Definicion de operacion compuesta por operandos y operadores */
struct OperacionCompuesta {
        Numero operandos<>;
        Operador operadores<>;
};
```

Representacion del programa y sus versiones

```
program CALCULADORA {
        version BASICA {
             ResultadoNumerico CALCULAR(Operacion) = 1;
        } = 1;
        version VECTORIAL {
             ResultadoVectorial CALCULAR(OperacionVectorial) = 1;
        } = 2;
        version COMPUESTA {
             ResultadoNumerico CALCULAR(OperacionCompuesta) = 1;
        } = 3;
} = 0x30005000;
```

Implementación y funcionamiento

Cliente

El programa cliente se encarga de solicitar al usuario el tipo de operacion a realizar y los datos que sean necesarios para procesar la información para poder enviar la solicitud al servidor empaquetando los datos recogidos en las estructuras previamente descritas

El programa interactua con el usuario mediante un menú interactivo con opciones segun el tipo de operacion que quiera calcular, peticiones de datos de entrada, mensajes de salida y mensajes de ayuda.

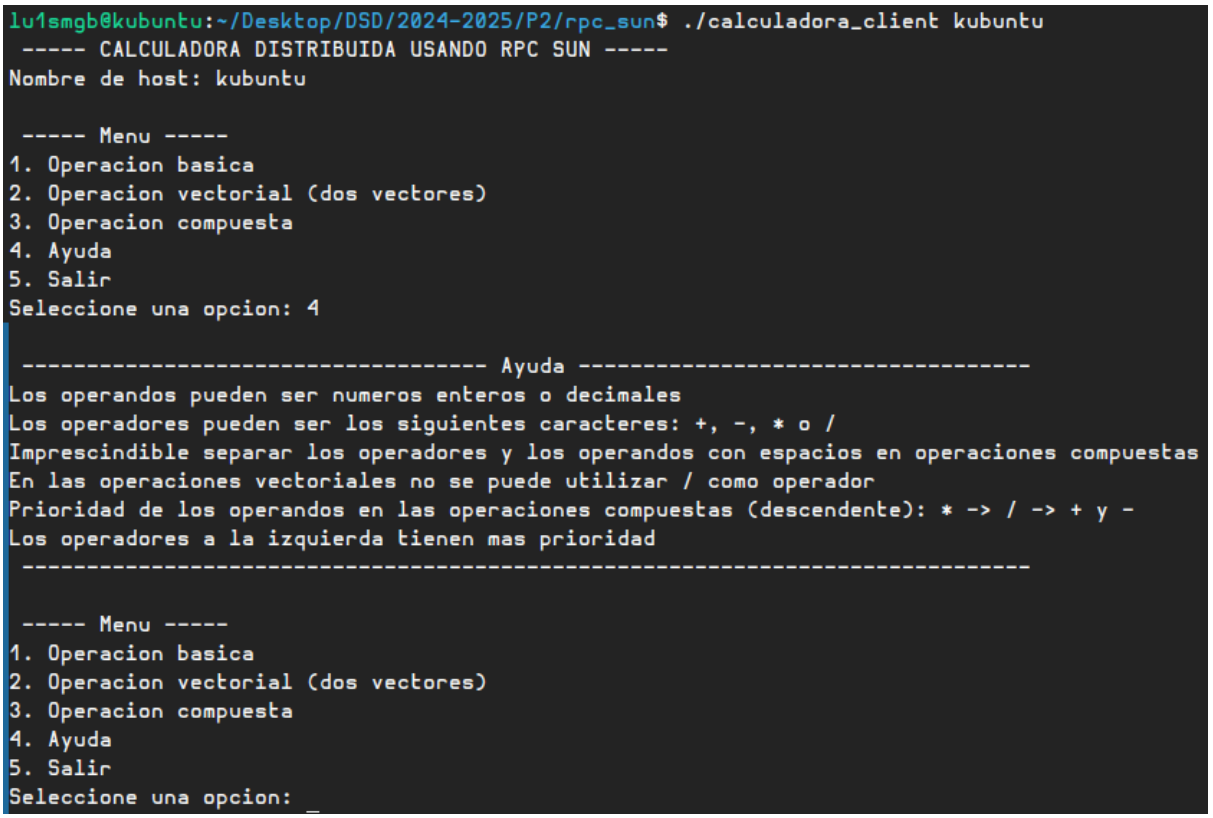
De forma general, el flujo de trabajo del programa cliente es el siguiente

1. Muestra el menu y pide al usuario una opcion por primera vez
2. Pide los datos que sean necesarios segun el tipo de operacion, junto con ayuda para no equivocarse durante la entrada de los datos
3. El cliente usa los datos recogidos para guardarlos en una estructura correspondiente y realiza una solicitud al servidor con esa estructura
4. Cuando recibe el resultado, lo muestra por pantalla y vuelve al menú

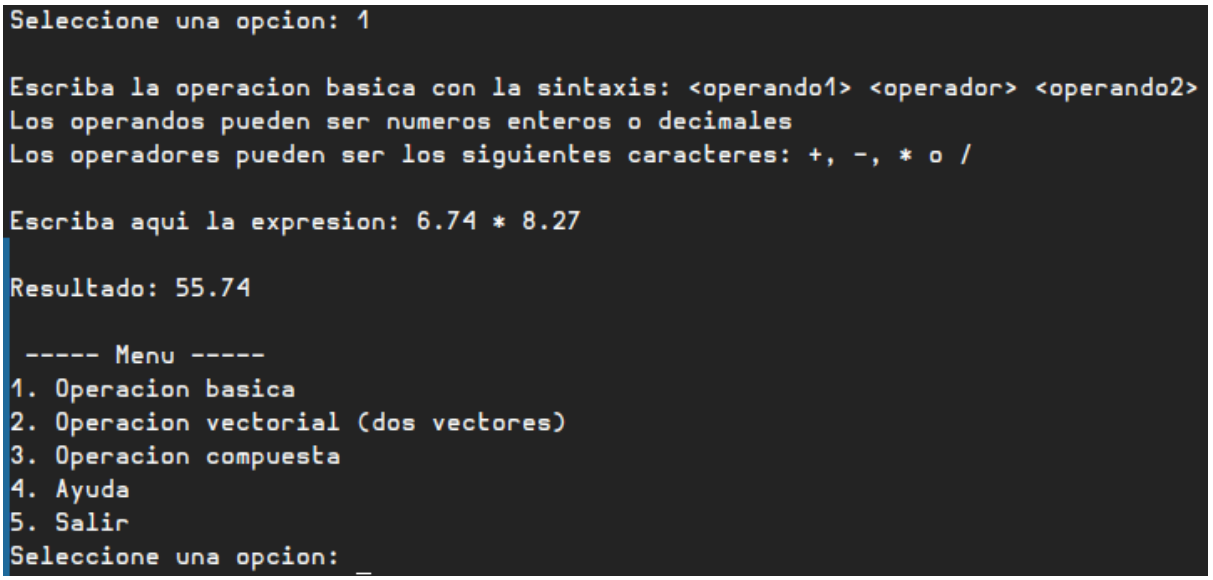
Para ejecutarlo tenemos que escribir en la terminal

```
./calculadora_client <host>
```

Donde host es el nombre o direccion IP del servidor



Operacion basica



Operacion vectorial

```
Seleccione una opcion: 2

Tamaño de los vectores (min 2, max 10): 3


Escribe los 3 operandos del primer vector
Los operandos pueden ser numeros enteros o decimales

Operando 1 de 3 del vector 1: 2
Operando 2 de 3 del vector 1: 4
Operando 3 de 3 del vector 1: 6


Escribe el operador
Los operadores pueden ser los siguientes caracteres: +, - o *
En las operaciones vectoriales no se puede utilizar / como operador

Operador: *


Escribe los 3 operandos del segundo vector
Los operandos pueden ser numeros enteros o decimales
Operando 1 de 3 del vector 2: 3
Operando 2 de 3 del vector 2: 6
Operando 3 de 3 del vector 2: 9

Lectura de usuario finalizada
[ 2.00 4.00 6.00 ] * [ 3.00 6.00 9.00 ]

Resultado: [ 6.00 24.00 54.00 ]

 ----- Menu -----
1. Operacion basica
2. Operacion vectorial (dos vectores)
3. Operacion compuesta
4. Ayuda
5. Salir
Seleccione una opcion: _
```

Operacion compuesta

```
Seleccione una opcion: 3

Escriba la operacion compuesta con la sintaxis:

<operando1> <operador1> <operando2> <operador2> <operando3> ... <operadorN> <operandoM>

Los operandos pueden ser numeros enteros o decimales
Los operadores pueden ser los siguientes caracteres: +, -, * o /
Imprescindible separar los operadores y los operandos con espacios
Prioridad de los operandos en las operaciones compuestas (descendente): * -> / -> + y -
Los operadores a la izquierda tienen mas prioridad

Expresion: 3 + 4 * 3 / 2 - 2

Operandos: 5
(3.00) (4.00) (3.00) (2.00) (2.00)
Operadores: 4
(+) (*) (/) (-)

Resultado: 7.00

 ----- Menu -----
1. Operacion basica
2. Operacion vectorial (dos vectores)
3. Operacion compuesta
4. Ayuda
5. Salir
Seleccione una opcion: _
```

Servidor

El servidor se encarga de recibir los datos de las operaciones enviadas por el cliente y producir y enviar un resultado de vuelta. Es el encargado de realizar todas las tareas aritméticas
Para ejecutarlo tenemos que escribir en la terminal

```
./calculadora_server
```

Operacion simple

El servidor simplemente calcula la solución según los dos operandos y el operador y lo devuelve al cliente

```
--- Se ha recibido una operacion SIMPLE ---

Calculando 6.74 * 8.27...
Resultado = 55.74

--- FIN. Enviando resultado a cliente ---
```

Operacion vectorial

El servidor calcula de forma simple para cada uno de los componentes de v1 y v2

```
--- Se ha recibido una operacion VECTORIAL ---

Componente 1
Calculando 2.00 * 3.00...
Resultado = 6.00

Componente 2
Calculando 4.00 * 6.00...
Resultado = 24.00

Componente 3
Calculando 6.00 * 9.00...
Resultado = 54.00

--- FIN. Enviando resultado a cliente ---
```

Operacion compuesta

El servidor sigue un algoritmo simple para el cálculo de este tipo de expresiones que, de manera resumida, sigue estos pasos

1. Encuentra la posición del operador más prioritario
2. Encuentra, en base a la posición del operador más prioritario, los dos operandos a los que afecta
3. Realiza el cálculo de la operación de los operandos y operador anteriores
4. Reemplaza los dos operandos por el resultado y elimina el operador utilizado de la estructura
5. Repite hasta que solo quede un operando en la estructura

```
--- Se ha recibido una operacion COMPUESTA ---

Operandos: 5
(3.00) (4.00) (3.00) (2.00) (2.00)
Operadores: 4
(+) (*) (/) (-)
Buscando *...
Encontrado operador * en posicion 1
Siguiente operador: *

Calculando 4.00 * 3.00...
Resultado = 12.00

Operandos: 4
(3.00) (12.00) (2.00) (2.00)
Operadores: 3
(+) (/) (-)
Buscando *...
Buscando /...
Encontrado operador / en posicion 1
Siguiente operador: /

Calculando 12.00 / 2.00...
Resultado = 6.00

Operandos: 3
(3.00) (6.00) (2.00)
Operadores: 2
(+) (-)
Buscando *...
Buscando /...
Buscando + o -...
Encontrado operador + en posicion 0
Siguiente operador: +

Calculando 3.00 + 6.00...
Resultado = 9.00

Operandos: 2
(9.00) (2.00)
Operadores: 1
(-)
Buscando *...
Buscando /...
Buscando + o -...
Encontrado operador - en posicion 0
Siguiente operador: -

Calculando 9.00 - 2.00...
Resultado = 7.00

Operandos: 1
(7.00)
Operadores: 0


--- FIN. Enviando resultado a cliente ---
```