

# Informe de Trabajo Práctico

## Nº1 - Parte C: Fundamentos - Herencia, agregación, polimorfismo

Programación Orientada a Objetos  
Ingeniería en Mecatrónica

Alumno: Juan Manuel BORQUEZ PEREZ  
Legajo: 13567



**UNCUYO**  
UNIVERSIDAD  
NACIONAL DE CUYO



**FACULTAD  
DE INGENIERÍA**

► **1983/2023**  
40 AÑOS DE DEMOCRACIA

# 1 Enunciado

- a) Prepare un directorio denominado `apellido_legajo_C` (reemplace con sus datos) que contenga 2 subdirectorios denominados `cons_4` y `cons_5`.
- b) Ubique en cada uno de ellos la implementación que corresponda según la consigna.
- c) Coloque en el directorio raíz (`apellido_legajo_C`) el documento con el informe general para esta entrega (siga el mismo formato de la anterior).
- d) Al finalizar, suba un archivo comprimido de la carpeta raíz creada (incluyendo todo su contenido) en la actividad denominada Entrega C, dentro del aula virtual.
- e) En cada consigna, Usted debe diseñar un modelo orientado a objetos que incorpore las clases mínimas indicadas e informarlo mediante un diagrama de clases.
- f) Fecha de entrega: 20 de setiembre de 2023.

## 1.1 Consigna 4

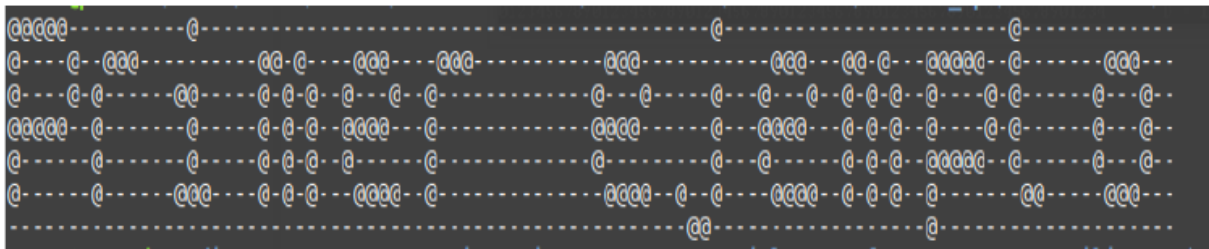
Implementar en lenguaje Python, bajo el paradigma orientado a objetos, un programa de utilidad que ofrezca un menú de usuario con la siguiente funcionalidad:

- Listar los archivos de arte ASCII disponibles en un directorio ubicado en el espacio de la aplicación.
- Seleccionar un archivo en particular mediante el tipeado de su nombre completo.
- Visualizar en la consola el contenido del archivo con el arte ASCII que contiene y un informe de análisis del mismo. Este informe debe indicar:
  - Qué caracteres se usaron en el archivo de arte ASCII.
  - Cuántos caracteres hay de cada uno y el total.
- Listar, en formato JSON, las coordenadas (solamente) que corresponden a un carácter indicado por el operador, considerando que `[0, 0]` se encuentra en la esquina superior izquierda.
- **[Desarrollo opcional]** Obtener la línea de texto del mensaje sin formato y su longitud, usando subprocesos y el binario del generador de arte ASCII.
- Terminar la ejecución.

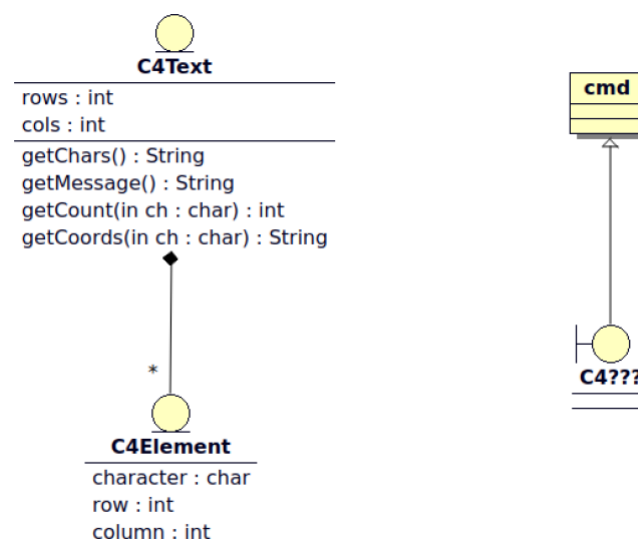
### 1.1.1 Observaciones y restricciones al diseño e implementación.

- Algunos archivos de prueba, para colocar dentro del directorio ya mencionado, se encuentran en el aula virtual (denominados `ejemploN.txt`), sin embargo, usted puede agregar otros si lo cree conveniente. Para ello, puede usar el generador provisto (denominado `c4ascii.cpp`).
- Los archivos de arte ASCII responden al siguiente formato interno:

- Contenido como texto plano.
- 7 filas de  $N$  caracteres ASCII, dependiendo  $N$  de la longitud de la línea con el mensaje.
- Un carácter de frente (para el texto del mensaje) y un carácter de fondo
- Por ejemplo:



- Actualmente, el generador admite 1 carácter de frente; sin embargo, se estima que su funcionalidad será ampliada para usar diferentes caracteres.
- Las opciones de visualización e informe no deben operar sobre el sistema de archivos.
- El aplicativo debe ofrecer mensajes de ayuda cuando ocurran situaciones no previstas o el usuario seleccione una opción que dependa de otra.
- Aplique un esquema de diseño que separe la capa de modelo de las de vista y control.
- Para el manejo de errores y excepciones, utilizar subclases de Exception.
- Para la interfaz de usuario, usar formato CLI y reutilización mediante herencia.
- Incluya en su modelo OO las siguientes clases, agregando aquellos elementos que considere adecuados:



## 1.2 Consigna 5

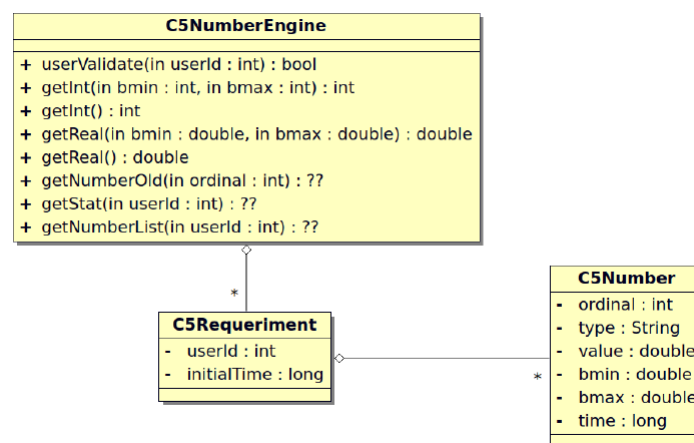
Utilizando UML para representar el modelo orientado a objetos y el lenguaje C++ para la implementación, desarrolle una pequeña herramienta para su uso en un sistema de comunicaciones en red conforme al requerimiento descripto:

- La aplicación deseada sigue una arquitectura cliente/servidor.
  - Los valores son puestos a disposición de los clientes mediante XML-RPC en un port específico para una IP dinámica, que se informa al momento del lanzamiento del servidor.
  - No se requiere una interfaz de usuario especial. Es una herramienta del tipo orden o comando del sistema(tanto para el servidor como para el cliente). Esto implica considerar el uso de argumentos de línea de comandos.
- El servidor debe proveer los siguientes servicios:
  - Validar cada petición en base a un identificador de usuario.
  - Permitir a un usuario iniciar un proceso de generación de números.
  - Generar un número real con 2 dígitos de precisión en un rango  $[-CI, CS]$ , donde CI y CS son los valores paramétricos de las cotas superior e inferior del rango, recibidos desde el cliente.
  - Generar un número entero en un rango  $[-CI, CS]$  proporcionado por el cliente.
  - Informar el valor de un número generado por el usuario, con su marca de tiempo y el rango asociados.
  - Informar la cantidad de números generados por el usuario, su suma total y su promedio.
  - Listar los números generados por el usuario remoto desde que inició el proceso de generación, incluyendo el tipo, el instante de tiempo en que se generó, el rango de cada uno y, al final, el tiempo que llevó realizar todo el proceso desde que se hizo la primera solicitud.
- El cliente debe permitir:
  - Solicitar un número entero en un rango indicado explícitamente.
  - Solicitar un número real en un rango indicado explícitamente.
  - Solicitar un número entero sin indicar el rango. En este caso, el servidor utiliza el mismo rango de la petición entera anterior.
  - Solicitar un número real sin especificar el rango. En este caso, el servidor utiliza el mismo rango de la petición real anterior.
  - Solicitar la información asociada a un número generado anteriormente. En este caso, la petición se realiza enviando el ordinal deseado.
  - Solicitar la estadística básica.
  - Solicitar el listado de números generados.

- Mostrar en pantalla cada respuesta obtenida.
- Guardar la respuesta obtenida en un archivo con formato XML bien formado. El nombre del archivo se provee en cada petición.

### 1.2.1 Observaciones y restricciones al diseño e implementación.

- Si bien son pocos usuarios, cada uno se identifica por un número de 3 dígitos.
- Los identificadores de los usuarios habilitados se encuentran almacenados en un archivo de texto predefinido del lado servidor.
- Un usuario puede acceder solo a los datos generados por él.
- La marca de tiempo de generación de cada número corresponde a la cantidad de segundos desde que el usuario inició el proceso.
- Contemplar el control de errores en los escenarios básicos de operación usando excepciones.
- Incluya los mensajes al usuario que correspondan, así como ayudas de operación o notificación de errores.
- Aplique un diseño de 2 capas con separación entre el modelo y la presentación/control.
- Por simplicidad, se sugiere el uso de la librería XML-RPC para C++, disponible desde la URL <http://xmlrpcpp.sourceforge.net/>. Considere la lectura de la documentación disponible en la página del proyecto. Revise el video explicativo de clase, sobre el montaje de un aplicativo C/S usando dicha librería, en <https://youtu.be/B4vzzSq0DvA> (aproximadamente desde 4:40 minutos). Puede ver un aplicativo similar en <https://youtu.be/2an54kpvg3c>. Sin embargo, queda a su preferencia el uso de una librería similar.
- Incluya en su modelo OO las siguientes clases, además de cualquier otro elemento que considere adecuado:



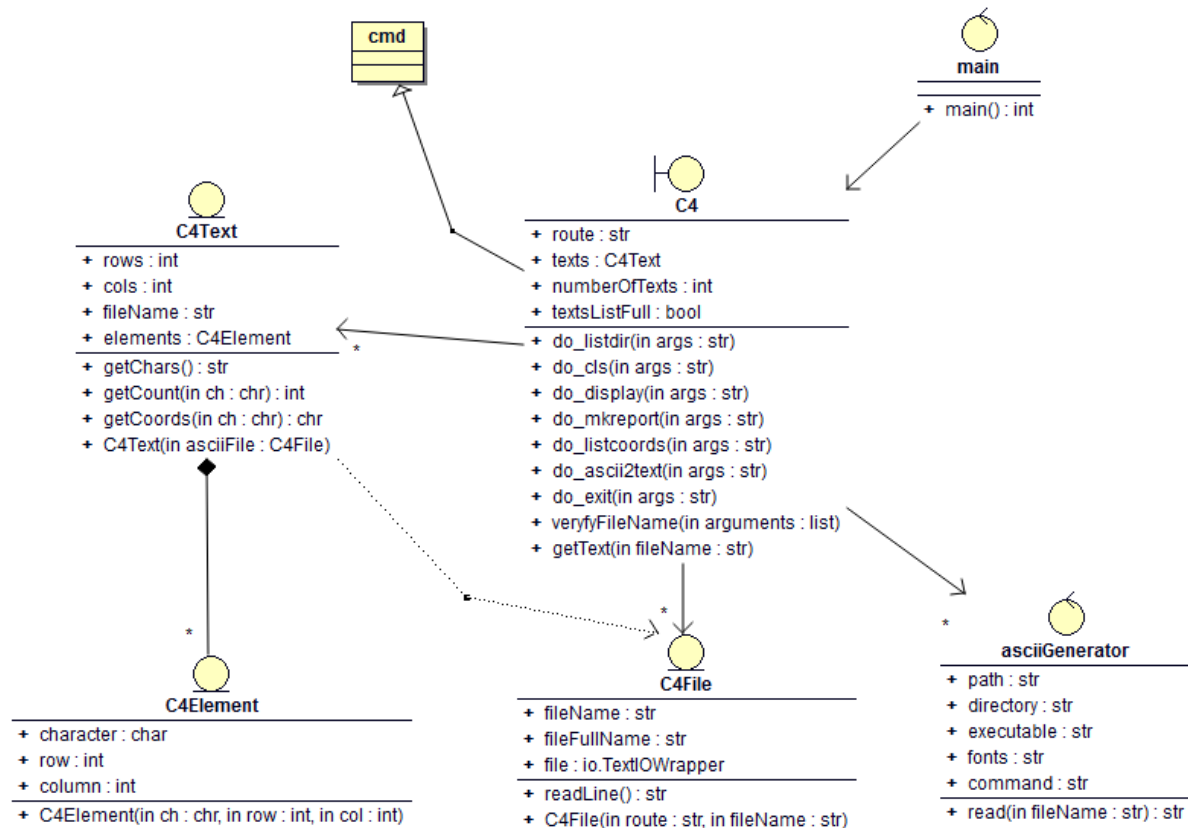
## 2 Esquema General de la Solución

### 2.1 Consigna 4

Esquema lógico general de la solución para la consigna 4.

La mayoría de la información relativa a las clases mostradas en el siguiente esquema está indicada como comentarios en los métodos y descripciones de las clases. A continuación, se realiza una descripción general de la solución:

- La clase ‘main’ representa el archivo principal de la solución.
- ‘C4’ es la principal clase de la solución que tiene los métodos CLI y permite la interacción con el usuario a través de prompts y salidas por pantalla.
- Los objetos de la clase ‘C4File’ están asociados a archivos de arte ASCII particulares. Se trata de objetos temporales utilizados únicamente durante la construcción de los objetos de la clase ‘C4Text’ para realizar la lectura de los archivos de arte ASCII.
- Los objetos de la clase ‘C4Text’ son representaciones del arte ASCII contenida en los archivos. Permiten acceder a información y operaciones generales de los mismos.
- Los objetos de la clase ‘C4Element’ son objetos persistentes contenidos dentro de objetos de la clase ‘C4Text’.
- ‘asciiGenerator’ es una clase con información y un método particular para el acceso al procedimiento remoto de la aplicación de ASCII art auxiliar. Tiene información de clase persistente común a todos los objetos de esta clase. Cada objeto se crea con el propósito de acceder al método.



## 2.2 Consigna 5

Esquema lógico general de la solución para la consigna 5

La mayoría de la información relativa a las clases mostradas en los siguiente esquema está indicada como comentarios en los métodos y descripciones de las clases. A continuación, se realiza una descripción general de la solución:

Se llevó a cabo un diagrama de clases para la parte del cliente y varios diagramas para la parte del servidor. Todos los diagramas y la distribución general de la aplicación, junto con las relaciones, se pueden acceder desde el proyecto de UML que se encuentra en la carpeta UML dentro de la consigna 5 en `./cons_5/C5/UML/C5`. El diagrama para la parte del cliente es menos cargado y se presenta en una sola captura en la figura 1, a la que se puede acceder desde el panel de navegación en BOUML como se indica en la captura 2.

La parte del servidor implica un conjunto de relaciones más complejas y en mayor cantidad si se tienen en cuenta todas las modificaciones realizadas. El diagrama completo para la parte del servidor no se presenta de forma legible en una sola captura (ver figura 3) y, por lo tanto, se decidió armar varios diagramas de clases para las partes más importantes del servidor. A estos diagramas se puede acceder desde el panel de navegación en UML como se indica en la captura 4.

Por un lado, en la figura 5 se encuentra el diagrama que representa las relaciones entre las clases principales dadas en la consigna. El diagrama que se muestra en las figuras 6 y 7 representa las relaciones entre clases asociadas a las excepciones personalizadas que se han definido. Las figuras 8 y 9 representan las relaciones entre clases asociadas al servidor

RPC.

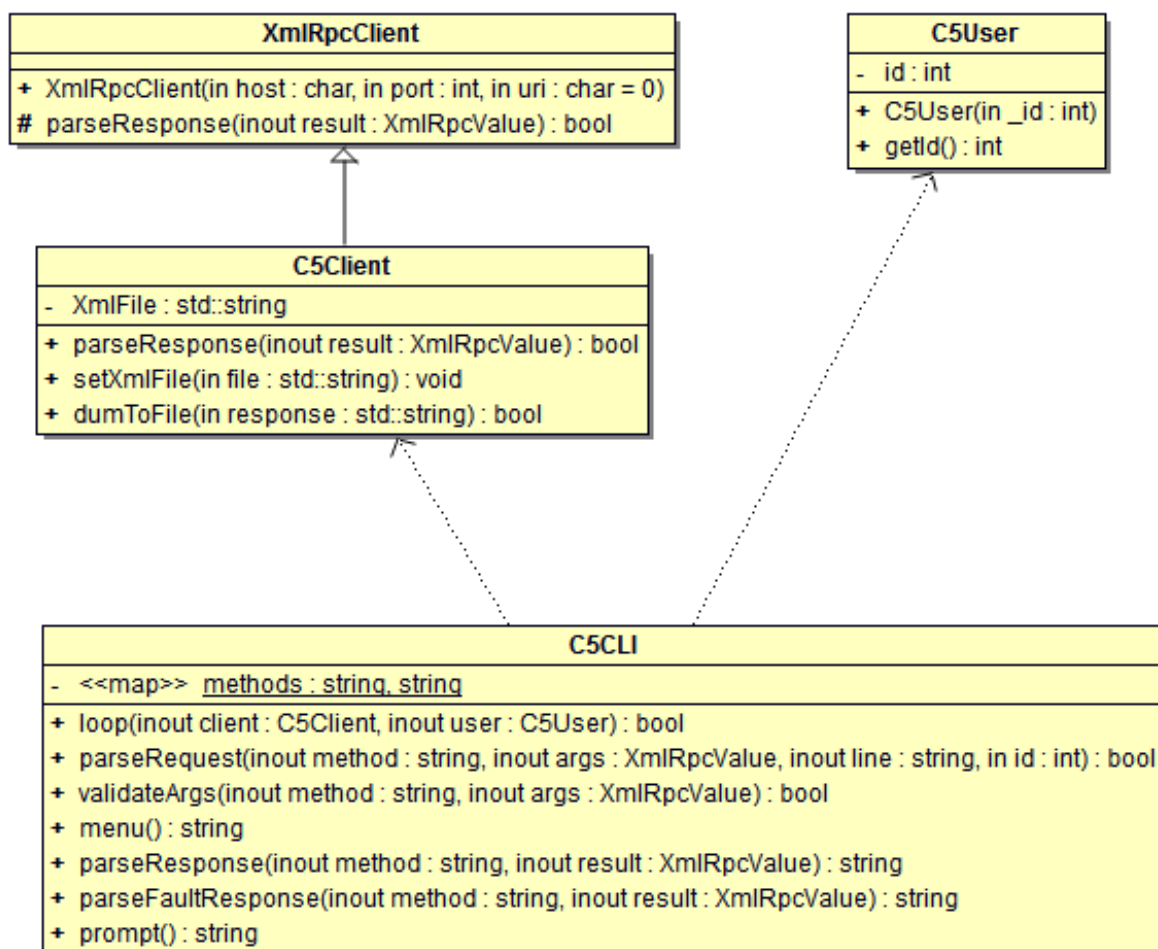


Figure 1: Diagrama de clases para la parte del cliente.



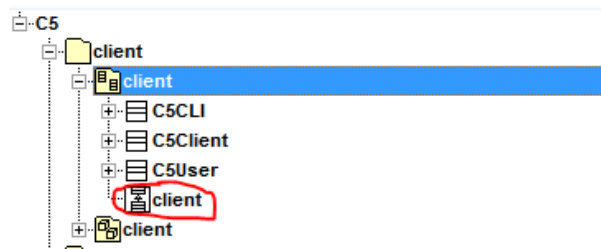


Figure 2: Captura del panel de navegación en BOUML para acceder al diagrama del cliente.

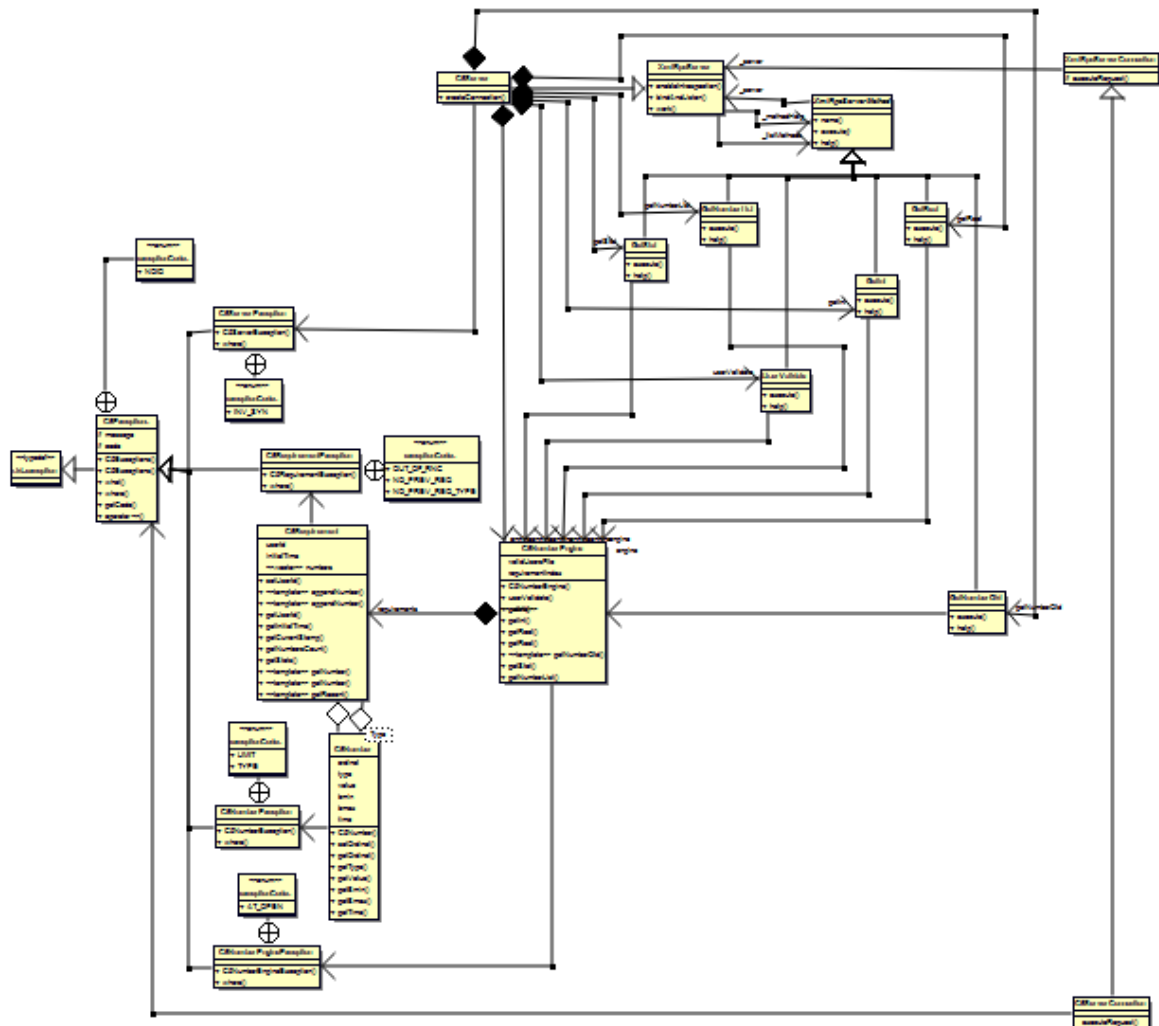


Figure 3: Diagrama completo para la parte del servidor.

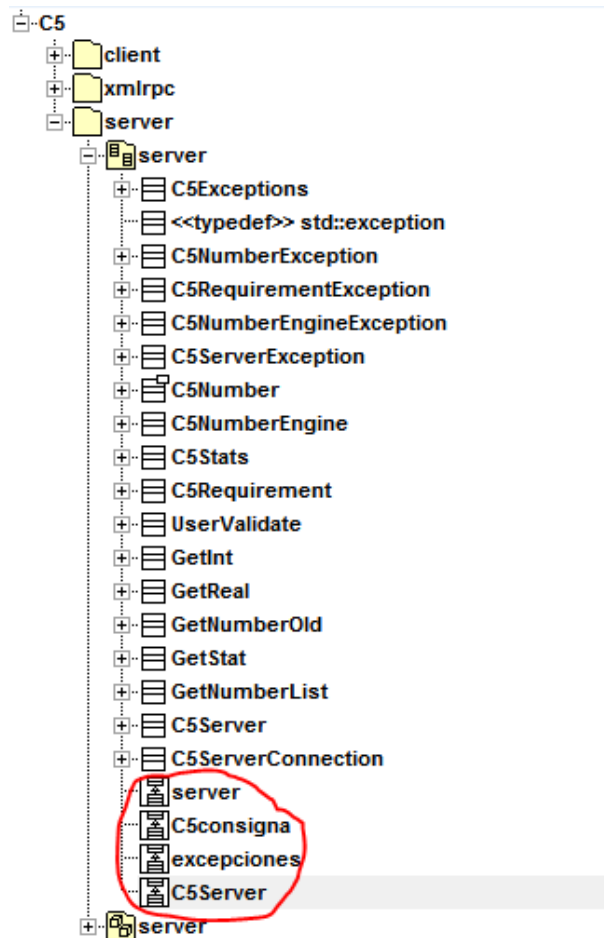


Figure 4: Captura del panel de navegación en UML para acceder a los diagramas del servidor.

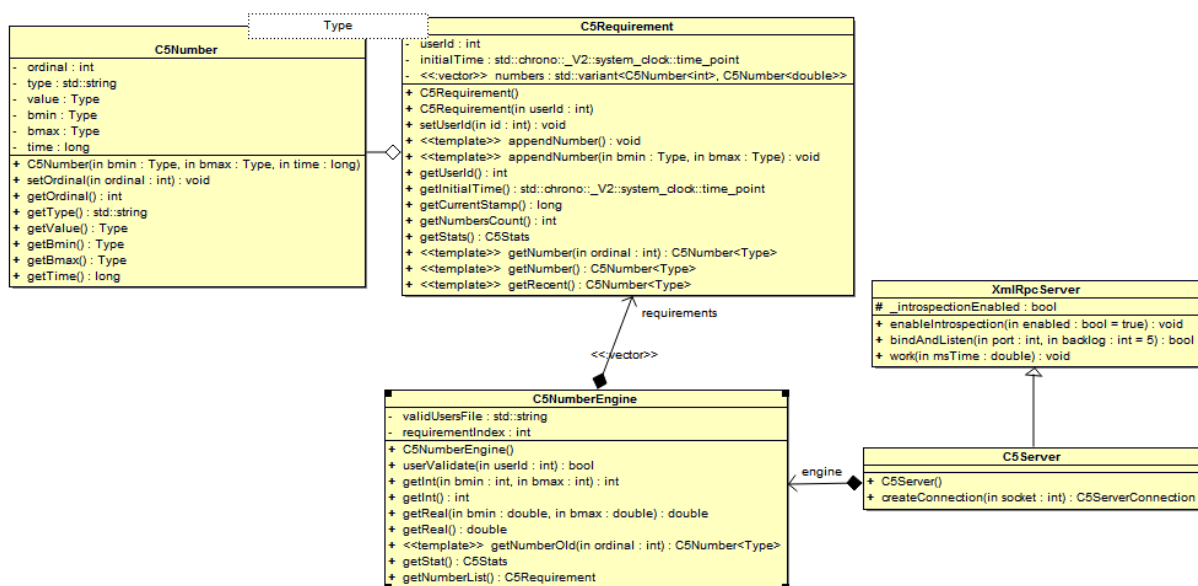


Figure 5: Diagrama de relaciones entre las clases principales.

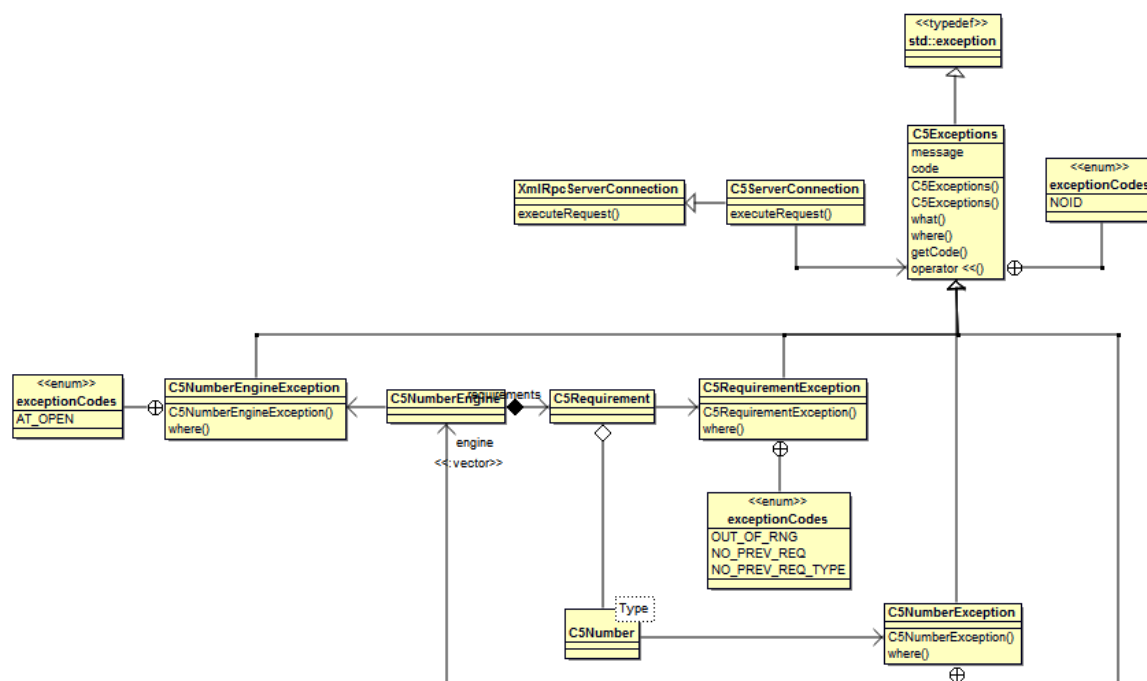


Figure 6: Diagrama de relaciones entre clases asociadas a excepciones personalizadas.

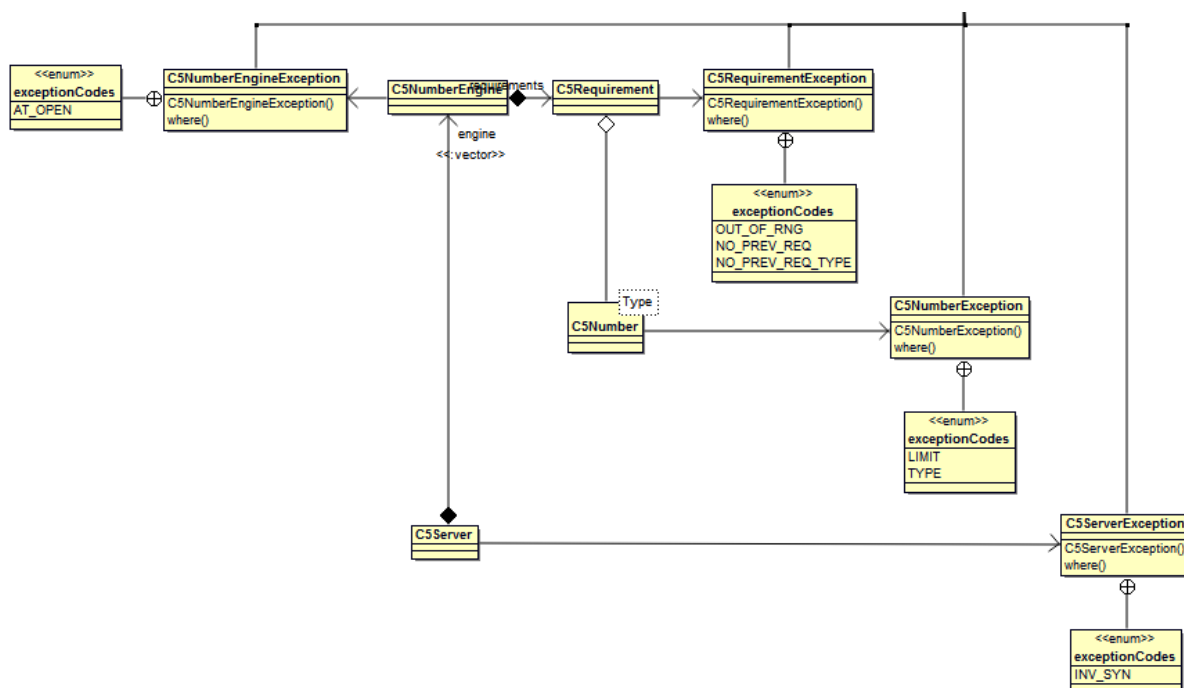


Figure 7: Diagrama de relaciones entre clases asociadas a excepciones personalizadas.

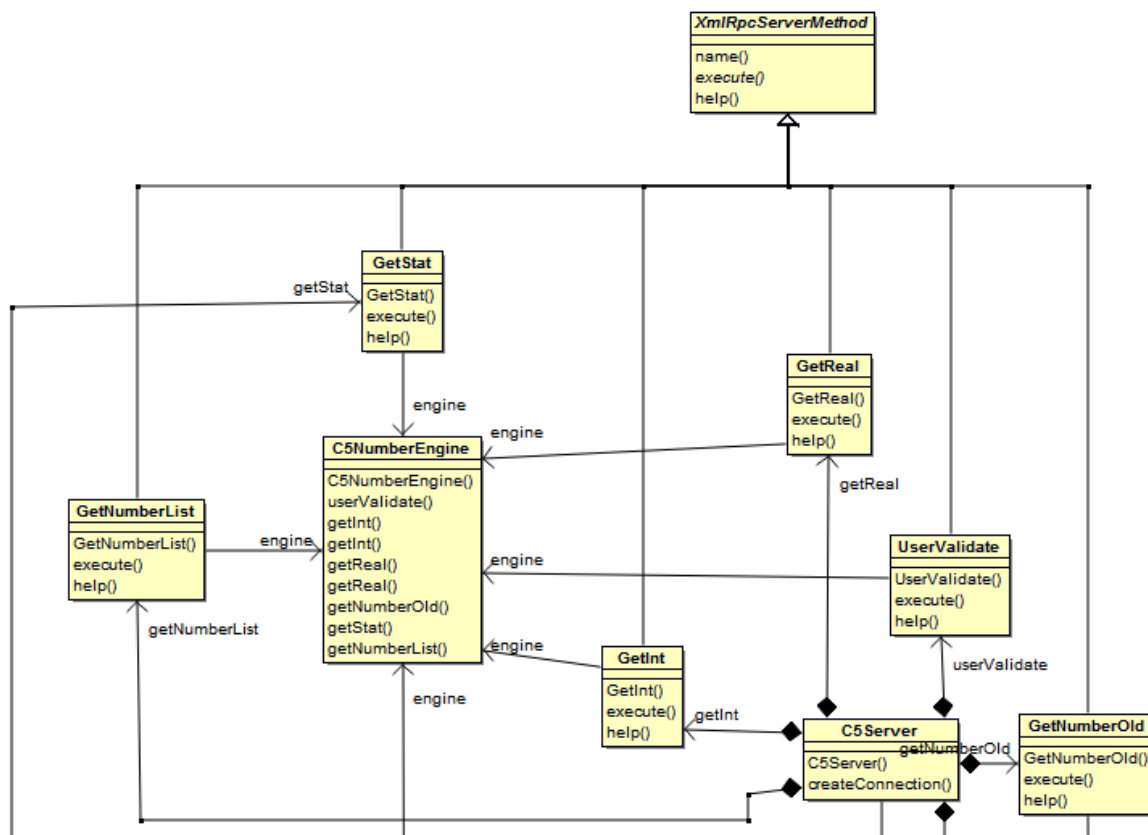


Figure 8: Diagrama de relaciones entre clases asociadas al servidor RPC.

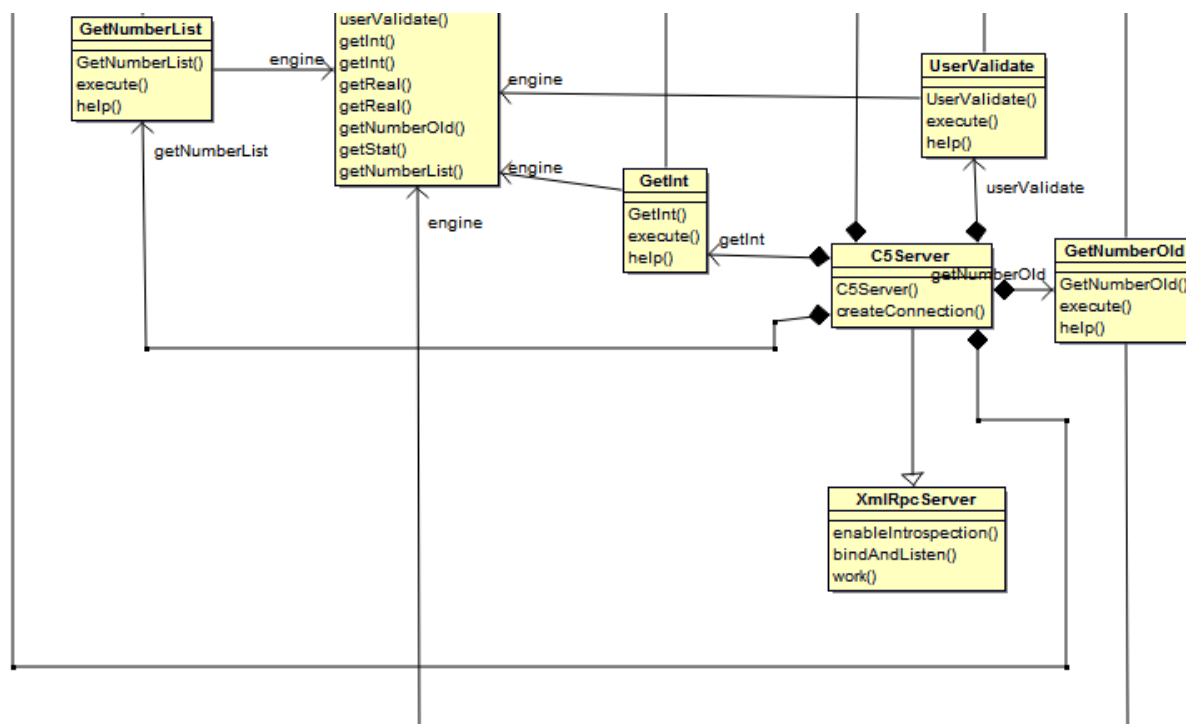
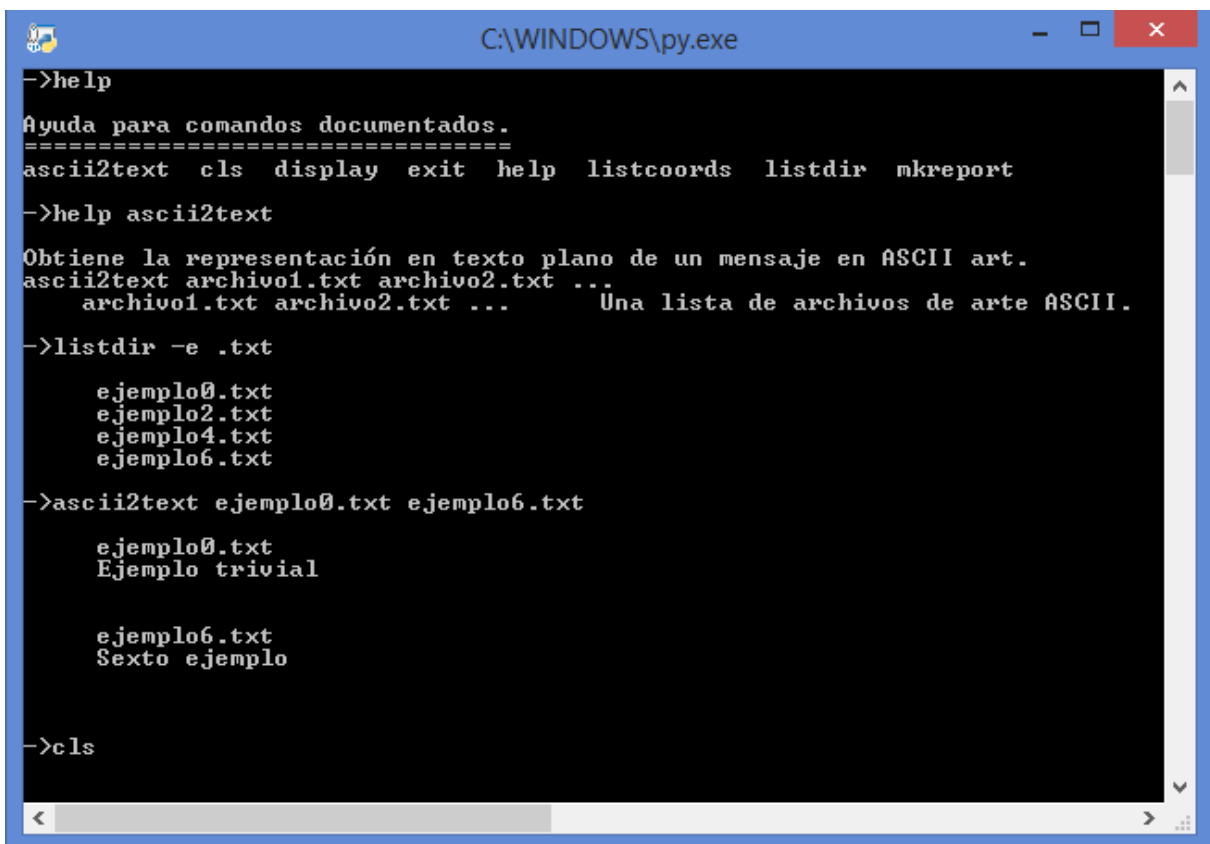


Figure 9: Diagrama de relaciones entre clases asociadas al servidor RPC.



```

C:\WINDOWS\py.exe
->help
Ayuda para comandos documentados.
=====
ascii2text cls display exit help listcoords listdir mkreport
->help mkreport
Realiza el reporte de los archivos de arte ASCII indicados.
mkreport archivo1.txt archivo2.txt ...
    archivo1.txt archivo2.txt ...    Una lista de archivos de arte ASCII.
->listdir -e .txt
    ejemplo0.txt
    ejemplo2.txt
    ejemplo4.txt
    ejemplo6.txt
->mkreport ejemplo2.txt ejemplo4.txt
+-----+
|          ejemplo2.txt          |
+-----+
| CARACTER | NUMERO DE OCURRENCIAS |
+-----+
|   e      |          554          |
|          |          181          |
+-----+
|  TOTAL   |          735          |
+-----+
+-----+
|          ejemplo4.txt          |
+-----+
| CARACTER | NUMERO DE OCURRENCIAS |
+-----+
|   -      |          534          |
|   #      |          152          |
+-----+
|  TOTAL   |          686          |
+-----+
->cls
  
```



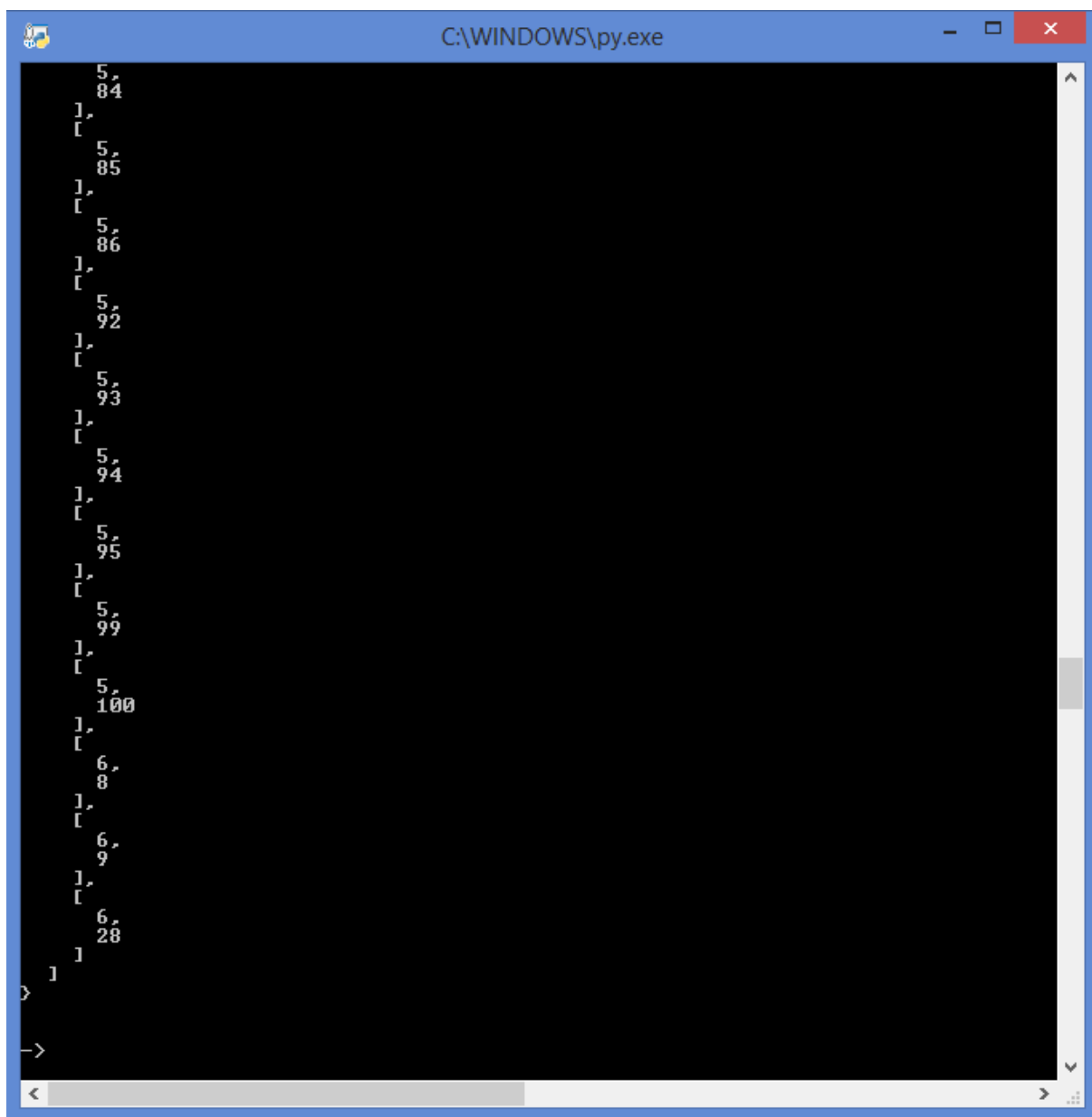
```
->help
Ayuda para comandos documentados.
=====
ascii2text  cls  display  exit  help  listcoords  listdir  mkreport
->help ascii2text
Obtiene la representación en texto plano de un mensaje en ASCII art.
ascii2text archivo1.txt archivo2.txt ...
        archivo1.txt archivo2.txt ...      Una lista de archivos de arte ASCII.
->listdir -e .txt
        ejemplo0.txt
        ejemplo2.txt
        ejemplo4.txt
        ejemplo6.txt
->ascii2text ejemplo0.txt ejemplo6.txt
        ejemplo0.txt
        Ejemplo trivial

        ejemplo6.txt
        Sexto ejemplo

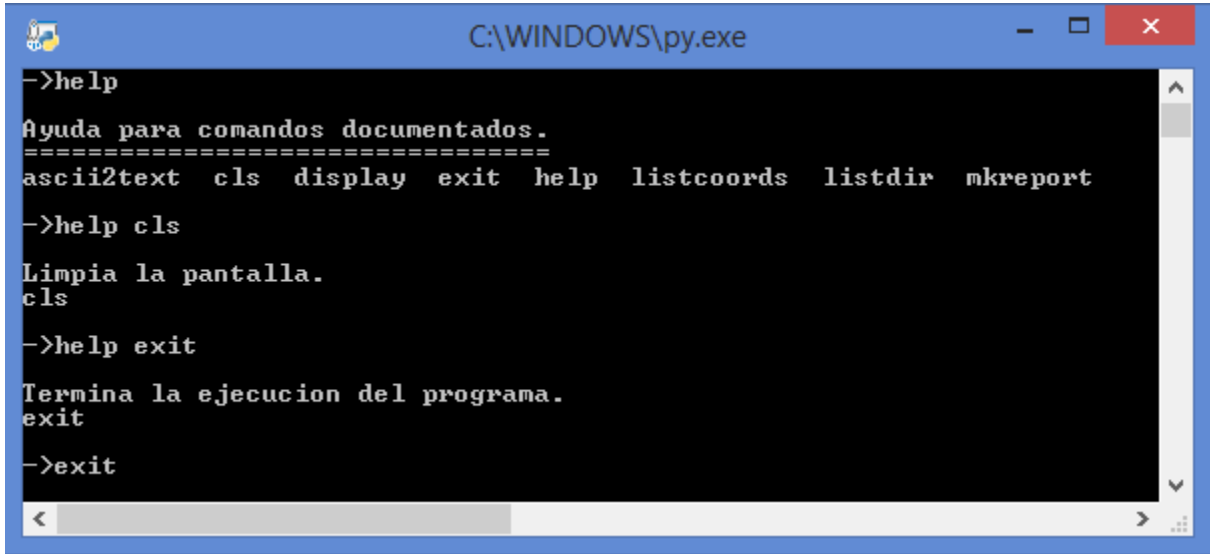
->cls
```

```
C:\WINDOWS\py.exe
->help
Ayuda para comandos documentados.
=====
ascii2text cls display exit help listcoords listdir mkreport
->help listcoords
Muestra las coordenadas en el archivo de un carácter indicado por el usuario.
listcoords 'ch' archivo1 archivo2 ...
    ch                               El caracter a buscar.
    archivo1.txt archivo2.txt ...    Una lista de archivos de arte ASCII.
->listdir -e .txt
    ejemplo0.txt
    ejemplo2.txt
    ejemplo4.txt
    ejemplo6.txt
->listcoords '#' ejemplo0.txt
    ejemplo0.txt
{
  "#": [
    [
      0,
      0
    ],
    [
      0,
      1
    ],
    [
      0,
      2
    ],
    [
      0,
      3
    ],
    [
      0,
      4
    ],
    [
      0,
      5
    ],
    [
      0,
      10
    ],
  ],
}
```





```
C:\WINDOWS\py.exe
5,
84
1,
[
5,
85
1,
[
5,
86
1,
[
5,
92
1,
[
5,
93
1,
[
5,
94
1,
[
5,
95
1,
[
5,
99
1,
[
5,
100
1,
6,
8
1,
6,
9
1,
6,
28
1
>
->
```

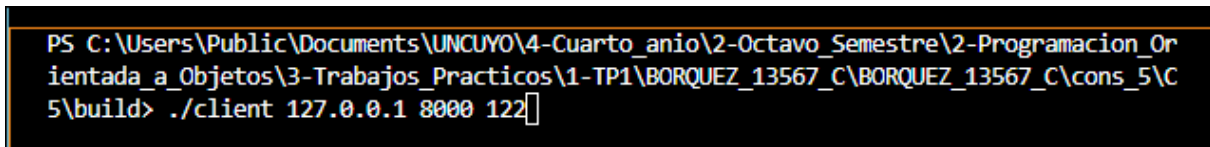


```
->help
Ayuda para comandos documentados.
=====
ascii2text  cls  display  exit  help  listcoords  listdir  mkreport
->help cls
Limpia la pantalla.
cls
->help exit
Termina la ejecucion del programa.
exit
->exit
```

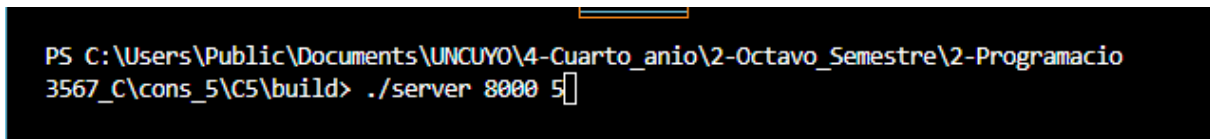
### 3.2 Consigna 5

Secuencia de Capturas para una ejecución completa y sin errores.

En las captura, en la parte derecha se muestra la ventana del cliente y en la parte izquierda la ventana del servidor. En este caso la verbosidad del servidor está seteada en su máximo valor (5).



```
PS C:\Users\Public\Documents\UNCUYO\4-Cuarto_anio\2-Octavo_Semestre\2-Programacion_Orientada_a_Objetos\3-Trabajos_Practicos\1-TP1\BORQUEZ_13567_C\BORQUEZ_13567_C\cons_5\C5\build> ./client 127.0.0.1 8000 122
```



```
PS C:\Users\Public\Documents\UNCUYO\4-Cuarto_anio\2-Octavo_Semestre\2-Programacio 13567_C\cons_5\C5\build> ./server 8000 5
```

```
PS C:\Users\Public\Documents\UNCUYO\4-Cuarto_año\2-Octavo_Semestre\2-Programación Orientada a Objetos\3-Trabajos Practicos\1-TP1\BORQUEZ_13567_C\BORQUEZ_13567_C\cons_5\C5\build> .\server 8000 5
Presione 'CTRL + C' para salir.
XmlRpcServer::bindAndListen: server listening on port 8000 fd 136
XmlRpcServer::work: waiting for a connection
```

```
PS C:\Users\Public\Documents\UNCUYO\4-Cuarto_año\2-Octavo_Semestre\2-Programación Orientada a Objetos\3-Trabajos Practicos\1-TP1\BORQUEZ_13567_C\BORQUEZ_13567_C\cons_5\C5\build> .\client 127.0.0.1 8000 122]
```

```
PS C:\Users\Public\Documents\UNCUYO\4-Cuarto_año\2-Octavo_Semestre\2-Programación Orientada a Objetos\3-Trabajos Practicos\1-TP1\BORQUEZ_13567_C\BORQUEZ_13567_C\cons_5\C5\build> .\server 8000 5
Presione 'CTRL + C' para salir.
XmlRpcServer::bindAndListen: server listening on port 8000 fd 136
XmlRpcServer::work: waiting for a connection
XmlRpcServer::acceptConnection: socket 140
XmlRpcServer::acceptConnection: creating a connection
XmlRpcServerConnection: new socket 140.
XmlRpcSocket::nbRead: read/recv returned 264.
XmlRpcSocket::nbRead: read/recv returned -1.
XmlRpcServerConnection::readHeader: read 264 bytes.
XmlRpcServerConnection::readHeader: specified content length is 148.
KeepAlive: 1
XmlRpcServerConnection::readRequest read 148 bytes.
XmlRpcServerConnection::executeRequest: server calling method 'userValidate'
XmlRpcServerConnection::generateResponse:
HTTP/1.1 200 OK
Server: XMLRPC++ 0.7
Content-Type: text/xml
Content-Length: 120

<?xml version="1.0"?>
<methodResponse><params><param>
  <value><i4>1</i4></value>
</param></params></methodResponse>

XmlRpcSocket::nbWrite: send/write returned 206.
XmlRpcServerConnection::writeResponse: wrote 206 of 206 bytes.
```

```
===== Comandos =====
- anterior
- entero
- estadística
- help
- listar
- real
- validar

Ingrese 'q' para salir.
Ingrese 'menu' para mostrar este menu.
Ingrese 'clear' para limpiar la pantalla.

Sintaxis de orden: 'comando' 'parametros separados por espacios.'
=====
>>> |
```

```
XmlRpcServerConnection::executeRequest: server calling method 'userValidate'
XmlRpcServerConnection::generateResponse:
HTTP/1.1 200 OK
Server: XMLRPC++ 0.7
Content-Type: text/xml
Content-Length: 120

<?xml version="1.0"?>
<methodResponse><params><param>
  <value><i4>1</i4></value>
</param></params></methodResponse>

XmlRpcSocket::nbWrite: send/write returned 206.
XmlRpcServerConnection::writeResponse: wrote 206 of 206 bytes.
XmlRpcSocket::nbRead: read/recv returned 274.
XmlRpcSocket::nbRead: read/recv returned -1.
XmlRpcServerConnection::readHeader: read 274 bytes.
XmlRpcServerConnection::readHeader: specified content length is 158.
KeepAlive: 1
XmlRpcServerConnection::readRequest read 158 bytes.
XmlRpcServerConnection::executeRequest: server calling method 'system.methodHelp'
XmlRpcServerConnection::generateResponse:
HTTP/1.1 200 OK
Server: XMLRPC++ 0.7
Content-Type: text/xml
Content-Length: 153

<?xml version="1.0"?>
<methodResponse><params><param>
  <value>Retrieve the help string for a named method</value>
</param></params></methodResponse>

XmlRpcSocket::nbWrite: send/write returned 239.
XmlRpcServerConnection::writeResponse: wrote 239 of 239 bytes.
```

```
===== Comandos =====
- anterior
- entero
- estadística
- help
- listar
- real
- validar

Ingrese 'q' para salir.
Ingrese 'menu' para mostrar este menu.
Ingrese 'clear' para limpiar la pantalla.

Sintaxis de orden: 'comando' 'parametros separados por espacios.'
=====
>>> help
respuesta:
  Retrieve the help string for a named method
>>> |
```

<pre> HTTP/1.1 200 OK Server: XMLRPC++ 0.7 Content-Type: text/xml Content-length: 153  &lt;?xml version="1.0"?&gt; &lt;methodResponse&gt;&lt;params&gt;&lt;param&gt;   &lt;value&gt;Retrieve the help string for a named method&lt;/value&gt; &lt;/param&gt;&lt;/params&gt;&lt;/methodResponse&gt;  XmlRpcSocket::nbWrite: send/write returned 239. XmlRpcServerConnection::writeResponse: wrote 239 of 239 bytes. XmlRpcSocket::nbRead: read/recv returned 269. XmlRpcSocket::nbRead: read/recv returned -1. XmlRpcServerConnection::readHeader: read 269 bytes. XmlRpcServerConnection::readHeader: specified content length is 153. KeepAlive: 1 XmlRpcServerConnection::readRequest read 153 bytes. XmlRpcServerConnection::executeRequest: server calling method 'system.methodHelp' XmlRpcServerConnection::generateResponse: HTTP/1.1 200 OK Server: XMLRPC++ 0.7 Content-Type: text/xml Content-length: 235  &lt;?xml version="1.0"?&gt; &lt;methodResponse&gt;&lt;params&gt;&lt;param&gt;   &lt;value&gt;Descripcion: valida al usuario segun su id.   Parametros: {id (int)}.   Returns: {validacion - (1 == valido)   (0 == no valido)}.&lt;/value&gt; &lt;/param&gt;&lt;/params&gt;&lt;/methodResponse&gt;  XmlRpcSocket::nbWrite: send/write returned 321. XmlRpcServerConnection::writeResponse: wrote 321 of 321 bytes. </pre>	<pre> ===== Comandos ===== - anterior - entero - estadistica - help - listar - real - validar  Ingrese 'q' para salir. Ingrese 'menu' para mostrar este menu. Ingrese 'clear' para limpiar la pantalla.  Sintaxis de orden: 'comando' 'parametros separados por espacios.' ===== &gt;&gt;&gt; help   respuesta:     Retrieve the help string for a named method  &gt;&gt;&gt; help validar   respuesta:     Descripcion: valida al usuario segun su id.     Parametros: {id (int)}.     Returns: {validacion - (1 == valido)   (0 == no valido)}.  &gt;&gt;&gt; </pre>
---	--

<pre> HTTP/1.1 200 OK Server: XMLRPC++ 0.7 Content-Type: text/xml Content-length: 235  &lt;?xml version="1.0"?&gt; &lt;methodResponse&gt;&lt;params&gt;&lt;param&gt;   &lt;value&gt;Descripcion: valida al usuario segun su id.   Parametros: {id (int)}.   Returns: {validacion - (1 == valido)   (0 == no valido)}.&lt;/value&gt; &lt;/param&gt;&lt;/params&gt;&lt;/methodResponse&gt;  XmlRpcSocket::nbWrite: send/write returned 321. XmlRpcServerConnection::writeResponse: wrote 321 of 321 bytes. XmlRpcSocket::nbRead: read/recv returned 339. XmlRpcSocket::nbRead: read/recv returned -1. XmlRpcServerConnection::readHeader: read 339 bytes. XmlRpcServerConnection::readHeader: specified content length is 223. KeepAlive: 1 XmlRpcServerConnection::readRequest read 223 bytes. XmlRpcServerConnection::executeRequest: server calling method 'getInt' XmlRpcServerConnection::generateResponse: HTTP/1.1 200 OK Server: XMLRPC++ 0.7 Content-Type: text/xml Content-length: 121  &lt;?xml version="1.0"?&gt; &lt;methodResponse&gt;&lt;params&gt;&lt;param&gt;   &lt;value&gt;&lt;i4&gt;10&lt;/i4&gt;&lt;/value&gt; &lt;/param&gt;&lt;/params&gt;&lt;/methodResponse&gt;  XmlRpcSocket::nbWrite: send/write returned 207. XmlRpcServerConnection::writeResponse: wrote 207 of 207 bytes. </pre>	<pre> ===== Comandos ===== - anterior - entero - estadistica - help - listar - real - validar  Ingrese 'q' para salir. Ingrese 'menu' para mostrar este menu. Ingrese 'clear' para limpiar la pantalla.  Sintaxis de orden: 'comando' 'parametros separados por espacios.' ===== &gt;&gt;&gt; help   respuesta:     Retrieve the help string for a named method  &gt;&gt;&gt; help validar   respuesta:     Descripcion: valida al usuario segun su id.     Parametros: {id (int)}.     Returns: {validacion - (1 == valido)   (0 == no valido)}.  &gt;&gt;&gt; entero 1 10   respuesta: 10 &gt;&gt;&gt; </pre>
--	---

```
XmlRpcServerConnection::executeRequest: server calling method 'getInt'
XmlRpcServerConnection::generateResponse:
HTTP/1.1 200 OK
Server: XMLRPC++ 0.7
Content-Type: text/xml
Content-length: 121

<?xml version="1.0"?>
<methodResponse><params><param>
  <value><i4>10</i4></value>
</param></params></methodResponse>

XmlRpcSocket::nbWrite: send/write returned 207.
XmlRpcServerConnection::writeResponse: wrote 207 of 207 bytes.
XmlRpcSocket::nbRead: read/recv returned 258.
XmlRpcSocket::nbRead: read/recv returned -1.
XmlRpcServerConnection::readHeader: read 258 bytes.
XmlRpcServerConnection::readHeader: specified content length is 142.
KeepAlive: 1
XmlRpcServerConnection::readRequest read 142 bytes.
XmlRpcServerConnection::executeRequest: server calling method 'getInt'
XmlRpcServerConnection::generateResponse:
HTTP/1.1 200 OK
Server: XMLRPC++ 0.7
Content-Type: text/xml
Content-length: 120

<?xml version="1.0"?>
<methodResponse><params><param>
  <value><i4>8</i4></value>
</param></params></methodResponse>

XmlRpcSocket::nbWrite: send/write returned 206.
XmlRpcServerConnection::writeResponse: wrote 206 of 206 bytes.
□
```

```
===== Comandos =====
- anterior
- entero
- estadística
- help
- listar
- real
- validar

Ingrese 'q' para salir.
Ingrese 'menu' para mostrar este menu.
Ingrese 'clear' para limpiar la pantalla.

 Sintaxis de orden: 'comando' 'parametros separados por espacios.'
=====
>>> help
  respuesta:
    Retrieve the help string for a named method

>>> help validar
  respuesta:
    Descripción: valida al usuario segun su id.
    Parametros: {id (int)}.
    Returns: {validacion - (1 == valido) | (0 == no valido)}.

>>> entero 1 10
  respuesta: 10
>>> entero
  respuesta: 8
>>> █
```

```
XmlRpcServerConnection::executeRequest: server calling method 'getInt'
XmlRpcServerConnection::generateResponse:
HTTP/1.1 200 OK
Server: XMLRPC++ 0.7
Content-Type: text/xml
Content-length: 120

<?xml version="1.0"?>
<methodResponse><params><param>
  <value><i4>8</i4></value>
</param></params></methodResponse>

XmlRpcSocket::nbWrite: send/write returned 206.
XmlRpcServerConnection::writeResponse: wrote 206 of 206 bytes.
XmlRpcSocket::nbRead: read/recv returned 372.
XmlRpcSocket::nbRead: read/recv returned -1.
XmlRpcServerConnection::readHeader: read 372 bytes.
XmlRpcServerConnection::readHeader: specified content length is 256.
KeepAlive: 1
XmlRpcServerConnection::readRequest read 256 bytes.
XmlRpcServerConnection::executeRequest: server calling method 'getReal'
XmlRpcServerConnection::generateResponse:
HTTP/1.1 200 OK
Server: XMLRPC++ 0.7
Content-Type: text/xml
Content-length: 136

<?xml version="1.0"?>
<methodResponse><params><param>
  <value><double>75.748278</double></value>
</param></params></methodResponse>

XmlRpcSocket::nbWrite: send/write returned 222.
XmlRpcServerConnection::writeResponse: wrote 222 of 222 bytes.
□
```

```
===== Comandos =====
- anterior
- entero
- estadística
- help
- listar
- real
- validar

Ingrese 'q' para salir.
Ingrese 'menu' para mostrar este menu.
Ingrese 'clear' para limpiar la pantalla.

 Sintaxis de orden: 'comando' 'parametros separados por espacios.'
=====
>>> help
  respuesta:
    Retrieve the help string for a named method

>>> help validar
  respuesta:
    Descripción: valida al usuario segun su id.
    Parametros: {id (int)}.
    Returns: {validacion - (1 == valido) | (0 == no valido)}.

>>> entero 1 10
  respuesta: 10
>>> entero
  respuesta: 8
>>> real 10 100.5
  respuesta: 75.75
>>> █
```

```
===== Comandos =====
- anterior
- entero
- estadística
- help
- listar
- real
- validar

Ingrese 'q' para salir.
Ingrese 'menu' para mostrar este menu.
Ingrese 'clear' para limpiar la pantalla.

Sintaxis de orden: 'comando' 'parametros separados por espacios.'
=====

>>> help
      respuesta:
          Retrieve the help string for a named method

>>> help valid
      respuesta:
          Descripción: valida al usuario según su id.
          Parametros: {id (int)}.
          Returns: {validacion - (1 == valido) | (0 == no valido)}.
```

```
Ingrese 'q' para salir.
Ingrese 'menu' para mostrar este menu.
Ingrese 'clear' para limpiar la pantalla.

 Sintaxis de orden: 'comando' 'parametros separados por espacios.'

>>> help
respuesta:
    Retrieve the help string for a named method

>>> help validar
respuesta:
    Descripcion: valida al usuario segun su id.
    Parametros: {id (int)}.
    Returns: {validacion - (1 == valido) | (0 == no valido)}.

>>> entero 10
respuesta:    10

>>> entero
respuesta:    8

>>> real 10 100.5
respuesta:    75.75

>>> real
respuesta:    71.34

>>> listar
respuesta:
```

valor	bmin	bmax	stamp(s)	tipo
10	1	10	154	entero
8	1	10	180	entero
75.75	10.00	100.50	213	real
71.34	10.00	100.50	238	real

Tiempo transcurrido(s): 264

```
</i4></value><value>i4>10</i4></value><value>154</value><value>entero</value></data></array></value><value><array><data><value>i4>8</i4></value><value>i4>1</i4></value><value>i4>10</i4></value><value>180</value><value>entero</value></data></array></value><value><array><data><value><double>75.748278</double></value><value><double>10.000000</double></value><value><double>100.500000</double></value><value><double>213</value><value>real</value></data></array></value><value><array><data><value><double>71.339538</double></value><value><double>10.000000</double></value><value><double>100.500000</double></value><value><double>238</value><value>real</value></data></array></value><value><double>264</value></data></param></params></methodResponse>
```

XmlRpcSocket::nbWrite: send/write returned 967.  
XmlRpcServerConnection::writeResponse: wrote 967 of 967 bytes.  
XmlRpcSocket::nbRead: read/recv returned 259.  
XmlRpcServerConnection::readHeader: read 259 bytes.  
XmlRpcServerConnection::readHeader: specified content length is 143.  
KeepAlive: 1  
XmlRpcServerConnection::readRequest read 143 bytes.  
XmlRpcServerConnection::executeRequest: server calling method 'getStat'  
XmlRpcServerConnection::generateResponse:  
HTTP/1.1 200 OK  
Server: XMLRPC++ 0.7  
Content-Type: text/xml  
Content-length: 246

```
<?xml version="1.0"?>
<methodResponse><params><param>
  <value><array><data><value>i4>4</i4></value><value><double>41.271954</double>
</value><value><double>165.087816</double></value></data></array></value>
</param></params></methodResponse>
```

XmlRpcSocket::nbWrite: send/write returned 332.  
XmlRpcServerConnection::writeResponse: wrote 332 of 332 bytes.

```
>>> help
respuesta:
  Retrieve the help string for a named method

>>> help validar
respuesta:
  Descripción: valida al usuario segun su id.
  Parametros: {id (int)}.
  Returns: {validacion - (1 == valido) | (0 == no valido)}.

>>> entero 1 10
respuesta: 10
>>> entero
respuesta: 8
>>> real 10 100.5
respuesta: 75.75
>>> real
respuesta: 71.34
>>> listar
respuesta:
```

valor	bmin	bmax	stamp(s)	tipo
10	1	10	154	entero
8	1	10	180	entero
75.75	10.00	100.50	213	real
71.34	10.00	100.50	238	real

Tiempo transcurrido(s): 264

```
>>> estadistica
respuesta:
```

cantidad	media	suma
4	41.27	165.09

```
</i4></value><value>i4>10</i4></value><value>154</value><value>entero</value></data></array></value><value><array><data><value>i4>8</i4></value><value>i4>1</i4></value><value>i4>10</i4></value><value>180</value><value>entero</value></data></array></value><value><array><data><value><double>75.748278</double></value><value><double>10.000000</double></value><value><double>100.500000</double></value><value><double>213</value><value>real</value></data></array></value><value><array><data><value><double>71.339538</double></value><value><double>10.000000</double></value><value><double>100.500000</double></value><value><double>238</value><value>real</value></data></array></value><value><double>264</value></data></param></params></methodResponse>
```

XmlRpcSocket::nbWrite: send/write returned 967.  
XmlRpcServerConnection::writeResponse: wrote 967 of 967 bytes.  
XmlRpcSocket::nbRead: read/recv returned 259.  
XmlRpcSocket::nbRead: read/recv returned -1.  
XmlRpcServerConnection::readHeader: read 259 bytes.  
XmlRpcServerConnection::readHeader: specified content length is 143.  
KeepAlive: 1  
XmlRpcServerConnection::readRequest read 143 bytes.  
XmlRpcServerConnection::executeRequest: server calling method 'getStat'  
XmlRpcServerConnection::generateResponse:  
HTTP/1.1 200 OK  
Server: XMLRPC++ 0.7  
Content-Type: text/xml  
Content-length: 246

```
<?xml version="1.0"?>
<methodResponse><params><param>
  <value><array><data><value>i4>4</i4></value><value><double>41.271954</double>
</value><value><double>165.087816</double></value></data></array></value>
</param></params></methodResponse>
```

XmlRpcSocket::nbWrite: send/write returned 332.  
XmlRpcServerConnection::writeResponse: wrote 332 of 332 bytes.

```
>>> real
respuesta: 75.75
>>> real
respuesta: 71.34
>>> listar
respuesta:
```

valor	bmin	bmax	stamp(s)	tipo
10	1	10	154	entero
8	1	10	180	entero
75.75	10.00	100.50	213	real
71.34	10.00	100.50	238	real

Tiempo transcurrido(s): 264

```
>>> estadistica
respuesta:
```

cantidad	media	suma
4	41.27	165.09

```
>>> menu
===== Comandos =====
- anterior
- entero
- estadistica
- help
- listar
- real
- validar

Ingrese 'q' para salir.
Ingrese 'menu' para mostrar este menu.
Ingrese 'clear' para limpiar la pantalla.

 Sintaxis de orden: 'comando' 'parametros separados por espacios.'
=====
```



```

XmlRpcSocket::nbWrite: send/write returned 332.
XmlRpcServerConnection::writeResponse: wrote 332 of 332 bytes.
XmlRpcSocket::nbRead: read/recv returned 338.
XmlRpcSocket::nbRead: read/recv returned -1.
XmlRpcServerConnection::readHeader: read 338 bytes.
XmlRpcServerConnection::readHeader: specified content length is 222.
KeepAlive: 1
XmlRpcServerConnection::readRequest read 222 bytes.
XmlRpcServerConnection::executeRequest: server calling method 'getInt'
XmlRpcServerConnection::generateResponse:
HTTP/1.1 200 OK
Server: XMLRPC++ 0.7
Content-Type: text/xml
Content-length: 120

<?xml version="1.0"?>
<methodResponse><params><param>
  <value><i4>6</i4></value>
</param></params></methodResponse>

XmlRpcSocket::nbWrite: send/write returned 206.
XmlRpcServerConnection::writeResponse: wrote 206 of 206 bytes.
XmlRpcSocket::nbRead: read/recv returned 338.
XmlRpcSocket::nbRead: read/recv returned -1.
XmlRpcServerConnection::readHeader: read 338 bytes.
XmlRpcServerConnection::readHeader: specified content length is 222.
KeepAlive: 1
XmlRpcServerConnection::readRequest read 222 bytes.
XmlRpcServerConnection::executeRequest: server calling method 'getInt'
Excepcion en CSNumber con codigo 0
Mensaje: Limite de numero invalido para el constructor
XmlRpcSocket::nbWrite: send/write returned 361.
XmlRpcServerConnection::writeResponse: wrote 361 of 361 bytes.

```

```

===== Comandos =====
- anterior
- entero
- estadística
- help
- listar
- real
- validar

Ingrese 'q' para salir.
Ingrese 'menu' para mostrar este menu.
Ingrese 'clear' para limpiar la pantalla.

 Sintaxis de orden: 'comando' 'parametros separados por espacios.'
=====
>>> entero 4 6
      respuesta: 6
>>> entero 6 4
      ocurrio una excepcion !!
      codigo - 0
      mensaje - Limite de numero invalido para el constructor
>>>

```

```

XmlRpcSocket::nbWrite: send/write returned 332.
XmlRpcServerConnection::writeResponse: wrote 332 of 332 bytes.
XmlRpcSocket::nbRead: read/recv returned 338.
XmlRpcSocket::nbRead: read/recv returned -1.
XmlRpcServerConnection::readHeader: read 338 bytes.
XmlRpcServerConnection::readHeader: specified content length is 222.
KeepAlive: 1
XmlRpcServerConnection::readRequest read 222 bytes.
XmlRpcServerConnection::executeRequest: server calling method 'getInt'
XmlRpcServerConnection::generateResponse:
HTTP/1.1 200 OK
Server: XMLRPC++ 0.7
Content-Type: text/xml
Content-length: 120

<?xml version="1.0"?>
<methodResponse><params><param>
  <value><i4>6</i4></value>
</param></params></methodResponse>

XmlRpcSocket::nbWrite: send/write returned 206.
XmlRpcServerConnection::writeResponse: wrote 206 of 206 bytes.
XmlRpcSocket::nbRead: read/recv returned 338.
XmlRpcSocket::nbRead: read/recv returned -1.
XmlRpcServerConnection::readHeader: read 338 bytes.
XmlRpcServerConnection::readHeader: specified content length is 222.
KeepAlive: 1
XmlRpcServerConnection::readRequest read 222 bytes.
XmlRpcServerConnection::executeRequest: server calling method 'getInt'
Excepcion en CSNumber con codigo 0
Mensaje: Limite de numero invalido para el constructor
XmlRpcSocket::nbWrite: send/write returned 361.
XmlRpcServerConnection::writeResponse: wrote 361 of 361 bytes.

```

```

===== Comandos =====
- anterior
- entero
- estadística
- help
- listar
- real
- validar

Ingrese 'q' para salir.
Ingrese 'menu' para mostrar este menu.
Ingrese 'clear' para limpiar la pantalla.

 Sintaxis de orden: 'comando' 'parametros separados por espacios.'
=====
>>> entero 4 6
      respuesta: 6
>>> entero 6 4
      ocurrio una excepcion !!
      codigo - 0
      mensaje - Limite de numero invalido para el constructor
>>> entero a b

- Sintaxis de comando incorrecta.

>>>

```





```
XmlRpcServerConnection::readHeader: read 338 bytes.
XmlRpcServerConnection::readHeader: specified content length is 222.
KeepAlive: 1
XmlRpcServerConnection::readRequest read 222 bytes.
XmlRpcServerConnection::executeRequest: server calling method 'getInt'
XmlRpcServerConnection::generateResponse:
HTTP/1.1 200 OK
Server: XMLRPC++ 0.7
Content-Type: text/xml
Content-Length: 120

<?xml version="1.0"?>
<methodResponse><params><param>
  <value><i4>6</i4></value>
</param></params></methodResponse>

XmlRpcSocket::nbWrite: send/write returned 206.
XmlRpcServerConnection::writeResponse: wrote 206 of 206 bytes.
XmlRpcSocket::nbRead: read/recvd returned 338.
XmlRpcSocket::nbRead: read/recvd returned -1.
XmlRpcServerConnection::readHeader: read 338 bytes.
XmlRpcServerConnection::readHeader: specified content length is 222.
KeepAlive: 1
XmlRpcServerConnection::readRequest read 222 bytes.
XmlRpcServerConnection::executeRequest: server calling method 'getInt'
Excepcion en CSNumber con codigo 0
Mensaje: Limite de numero invalido para el constructor
XmlRpcSocket::nbWrite: send/write returned 361.
XmlRpcServerConnection::writeResponse: wrote 361 of 361 bytes.
XmlRpcSocket::nbRead: read/recvd returned -1.
XmlRpcSource::close: closing socket 140.
XmlRpcSocket::close: fd 140.
XmlRpcSource::close: done closing socket 140.
XmlRpcSource::close: deleting this
XmlRpcServerConnection dtor.
```

```
Comandos
- anterior
- entero
- estadística
- help
- listar
- real
- validar

Ingrese 'q' para salir.
Ingrese 'menu' para mostrar este menu.
Ingrese 'clear' para limpiar la pantalla.

Sintaxis de orden: 'comando' 'parametros separados por espacios.'

>>> entero 4 6
      respuesta: 6
>>> entero 6 4
      ocurrió una excepcion !!
      codigo - 0
      mensaje - Limite de numero invalido para el constructor
>>> entero a b

- Sintaxis de comando incorrecta.

>>> q

- Hasta luego!

PS C:\Users\Public\Documents\UNCUYO\4-Cuarto_año\2-Octavo_Semestre\2-Programación Orientada a Objetos\3-Trabajos_Practicos\1-TP1\BORQUEZ_13567_C\BORQUEZ_13567_C\cons_5\C5\build>
```

## 4 Uso

### 4.1 Consigna 4

La aplicación se ejecuta abriendo el archivo "main.py".

La aplicación asociada a esta consigna permite el tratamiento general de archivos con arte ASCII. Presenta una interfaz de línea de comandos que permite al usuario llevar a cabo distintas operaciones mediante comandos.

Las operaciones que se pueden realizar con archivos de arte ASCII están documentadas. Puedes acceder a esta documentación de ayuda para cada comando utilizando el comando "help" sin parámetros, que lista todos los comandos disponibles, o utilizando el mismo comando seguido del nombre de un comando específico para acceder a ayuda específica para ese comando ("help comando").

A continuación, se indican las operaciones que se pueden llevar a cabo:

- **listdir**: Muestra los nombres de los archivos en el directorio de la aplicación. También puedes listar aquellos con una extensión específica, por ejemplo, .txt para arte ASCII.
- **display**: Muestra por pantalla el contenido de una lista de archivos de arte ASCII.
- **mkreport**: Genera un reporte del contenido del archivo de arte ASCII, incluyendo los caracteres utilizados y el número de ocurrencias.
- **listcoords**: Muestra las coordenadas en las que se presenta un carácter específico dentro de un archivo de arte ASCII.
- **ascii2text**: Convierte el contenido de un archivo de arte ASCII en un mensaje en texto plano.
- **cls**: Limpia la pantalla.

- **exit**: Termina la ejecución.

Para que la aplicación funcione correctamente, es importante que todos los archivos de código se encuentren en el mismo directorio tal como se entregan. Además, el módulo auxiliar en C++ utilizado para el tratamiento de archivos de arte ASCII, el cual se usa para la traducción del contenido de arte ASCII a un mensaje en texto plano, debe estar contenido en la carpeta especificada dentro del directorio de la solución, junto con el archivo "font.txt". En resumen, la disposición de archivos en directorios debe ser la misma que se proporciona con la aplicación.

El manejo de errores se realiza principalmente en los métodos de comandos de la clase C4. En los demás módulos, se lanzan excepciones en casos específicos. Aprovechamos el sistema para la validación de archivos, por lo que no realizamos explícitamente una verificación de la existencia de un archivo. En su lugar, al intentar abrir un archivo que no existe, capturamos la excepción generada por el sistema.

## 4.2 Consigna 5

Una explicación detallada sobre el uso de la aplicación se puede encontrar en el archivo `README.txt` ubicado en la carpeta de la solución en `cons_5/C5`. Del mismo modo, durante la ejecución, se proporcionan instrucciones de ayuda al usuario que utiliza la interfaz de línea de comandos (CLI).

En relación con la extensión del código o el acceso a especificaciones más detalladas sobre el código fuente y su implementación, existe un archivo de configuración de documentación llamado 'DoxyFile' en la carpeta 'doc' en `cons_5/C5`. Puede utilizarse este archivo para llevar a cabo la documentación automática del código. Los archivos de documentación se generarán en la misma carpeta 'doc'. Posteriormente, es posible abrir la carpeta con un visor de HTML, como la extensión Live Server para VSCode, para visualizar la documentación de las clases, métodos, etc.

Para generar la documentación, es necesario ejecutar el siguiente comando desde una terminal en donde se encuentra el archivo de configuración (es necesario tener doxygen instalado):

```
doxygen DoxyFile.
```

## 5 Conclusiones

### 5.1 Cambios Realizados/Particularidades

#### 5.1.1 Consigna 4

La mayoría de los cambios realizados en el código se encuentran indicados en el código fuente de la aplicación como comentarios de línea. Otros de los cambios realizados son los siguientes.

En C4Text:

**Atributos:**

- **fileName**: El nombre del archivo de arte ASCII al que está asociado.

- **elements**: Una matriz (lista de listas) de elementos C4Elements.

#### **Métodos:**

- **getChars**: Devuelve una lista con los caracteres en el archivo en lugar de un string.
- **getCoord**: Devuelve un diccionario en el que la clave es el carácter indicado y el valor es una lista con las coordenadas (tuplas) de ocurrencia del carácter.
- **C4Text**: El constructor para objetos de la clase recibe el nombre de un archivo de arte ASCII. En el constructor se obtiene la lista de elementos asociados al objeto.
- **str**: Permite la obtener la representación del objeto en una cadena de caracteres.

#### **En C4:**

- No se dispone de comandos específicos para seleccionar archivos o conjuntos de archivos en esta clase. En su lugar, los comandos pueden aplicarse en cualquier momento a cualquiera de los archivos de arte ASCII que se encuentran en el directorio. Esto se hizo para hacer más cómodo para el usuario el uso de la aplicación, en lugar de requerir una selección previa de archivos para operar.
- El manejo de errores durante la ejecución se aplica en los métodos de esta clase, aunque se limita a mostrar mensajes de error en relación a las excepciones sin realizar, por ejemplo, operaciones de seguimiento de errores.
- Para reducir la cantidad de veces que se abre un archivo de arte ASCII y se crea el objeto C4Text asociado, se incluye como atributo de esta clase una colección (lista) de objetos C4Text creados durante las llamadas a comandos. De esta manera, cuando un comando requiere la creación de un objeto C4Text para un archivo que se abrió recientemente, en lugar de crear un nuevo objeto asociado a ese archivo, se accede directamente a él desde la lista si está presente o se crea si no lo está. La lista tiene un tamaño máximo de 5 elementos. Cuando se llena, se agrega un nuevo elemento al final de la lista y se elimina el primero. Para identificar la asociación de objetos C4Text con archivos de arte ASCII y poder recuperarlos de la lista, se incluyó como atributo adicional el nombre del archivo al que están asociados en los objetos de la clase C4Text.
- Algunos comandos, como el de limpieza de pantalla y el de visualización de archivos, hacen uso de comandos específicos del sistema. Dado que la aplicación puede ejecutarse en diferentes sistemas operativos, se verifica la plataforma antes de ejecutar esos comandos. En principio, la funcionalidad debería estar disponible en Windows, Linux y macOS.

### **5.1.2 Consigna 5**

La mayoría de los cambios realizados en el código se encuentran indicados en el código fuente de la aplicación como comentarios de línea. A continuación se puede mencionar algunas de las particularidades de la implementación.

#### **Uso de Templates:**

- La clase C5Number se define como un template para abarcar las dos posibilidades de un objeto almacenando un número entero o un número real correspondientes a un tipo de dato int o double respectivamente.
- Debido al uso de los templates, algunas funciones son implementadas en archivos con extensión t.hpp. De echo, al archivo C5Number.h no hay asociado un archivo .cpp pero sí un archivo .t.hpp.
- Dado que existen en el modelo otras clases que hacen uso de los métodos de los objetos de la clase C5Number, para estas también se pueden encontrar archivos .t.hpp donde se implementan métodos template.

Uso de **Variants**:

- Se usaron variants para la lista (vector) de números para cada requerimiento. Cada objeto variant entonces puede almacenar bien un C5Number de tipo entero o un C5Number de tipo double.

Herencia de clases propias de la la libreria de **XmlRpc**.

- En algunas oportunidades, algunas clases se definieron heredando de clases propias en la libreria Xmlrpc con el objeto principal de sobrescribir métodos para personalizarlos a la aplicación.
- En concreto, esto se hace en la definición del C5Server y otras clases asociadas a los XmlRpcServer.
- Por ejemplo, en una de las ocasiones se hizo esto con el objeto de hacer el manejo de las excepciones propias definidas y el envío de los mensajes de error asociados.
- En estos casos, el resto de la funcionalidad que aporta la libreria es aprovechada.

Ayuda para los métodos RPC en el servidor. En lugar de implementarse la ayuda para los métodos RPC en el cliente RPC se optó por aprovechar una funcionalidad ya aportada en la libreria de XmlRpc para brindar ayuda de los métodos. En concreto, se habilitó la introspección del servidor para la aceptación de llamadas remotas a métodos de ayuda para los métodos RPC. Para hacer uso de esta funcionalidad se definieron, para cada objeto de XmlRpcServerMethod un método de ayuda ya incorporado en la declaración de al clase con la calificación virtual, lo que indica que en principio no tienen implementación.

## 5.2 Dificultades Encontradas

Se intentó hacer multithreading para el funcionamiento del servidor en paralelo con una función que espera ala presión de alguna tecla para terminar con la ejecución de modo que al presionar el boton se llame a un método del server que permite la desconexión controlada del mismo. Como no se pudo lograr, la aplicación del server se termina con CTRL + C.

### 5.3 Comentario/Extensiones

- En algunas partes del código para el acceso a los objetos C5Number en la lista de números de cada requerimiento C5Requirement, se hace uso de una excepción bad variant access para determinar el tipo del C5Number (entero o real) que contiene un variant en una posición de la lista de requerimientos. Si bien esto sirve, tal vez no sea la opción más adecuada para hacer el acceso a los elementos de la lista de números del requerimiento. Tal vez la opción no sea tan descabellada en este caso en el que los objetos variant a lo sumo pueden guardar dos tipos de dato, pero en otros casos en los que se tengan mas tipos de dato esta alternativa no es buena.
- Se podría utilizar otro tipo de extensión para el archivo de usuarios registrados.
- No se tiene un manejo de excepciones propios para la parte del cliente, que en caso de ser necesario habría que agregar.

### 5.4 Valoración Personal/Observaciones

Considero que he logrado cumplir con los requisitos establecidos en las consignas aplicando los conceptos de Programación Orientada a Objetos. Entre los aspectos que destacaría:

[5], [4], [3], [1], [6], [2]

## Bibliografía

- [1] *C++ FAQ: Templates*. URL: <https://isocpp.org/wiki/faq/templates>.
- [2] ChatGPT. *OpenAI GPT-3.5*. Personal Assistant AI. URL: <https://www.openai.com/>.
- [3] CodeProject. *How to Define a Template Class in a .h File and Implement in a .cpp File*. URL: <https://www.codeproject.com/Articles/48575/How-to-Define-a-Template-Class-in-a-h-File-and-Impl>.
- [4] Microsoft. *Source Code Organization: C++ Templates*. URL: <https://learn.microsoft.com/en-us/cpp/cpp/source-code-organization-cpp-templates?view=msvc-170>.
- [5] *What is the difference between a template class and a class template?* URL: <https://stackoverflow.com/questions/879535/what-is-the-difference-between-a-template-class-and-a-class-template>.
- [6] *Why can templates only be implemented in the header file?* URL: <https://stackoverflow.com/questions/495021/why-can-templates-only-be-implemented-in-the-header-file/495056#495056>.