

Nombre del método	Requerimiento o asociado	Precondición	Postcondición	Modelo verbal
Controlador(Vista& pVista)	CTL-1	Existe una instancia de Vista válida	Controlador inicializado con referencia a la vista	Inicializar el controlador recibiendo la referencia a la vista para manejar la interacción.
obtenerEleccionMenu() const	CTL-2	Mostrar menú disponible en vista	Retorna la elección del usuario	Solicitar a la vista la elección de menú y devolverla al flujo principal.
obtenerAccionJugador() const	CTL-3	Turno activo de un jugador	Retorna la acción seleccionada por el jugador	Pedir a la vista la acción del jugador (HIT/STAND) y retornarla para procesamiento .
Apuesta()	AP-1	Datos iniciales del jugador disponibles	Objeto Apuesta inicializado	Crear la instancia de Apuesta con valores por defecto.
aumentarApuesta(double cantidad)	AP-2	Saldo suficiente para aumentar	Apuesta incrementada y saldo ajustado	Incrementar la apuesta actual en la cantidad indicada si hay saldo suficiente.
reducirApuesta(double cantidad)	AP-3	Existe una apuesta	Apuesta reducida y saldo	Reducir la apuesta en la cantidad

		activa	ajustado	indicada y devolver saldo si aplica.
resetearApuesta()	AP-4	Apuesta modificable	Apuesta en valor cero o inicial	Restablecer la apuesta al valor inicial por defecto.
ganar()	AP-5	Apuesta activa y resultado ganador	Saldo actualizado con pago	Aplicar la ganancia correspondiente al jugador por la apuesta ganada.
perder()	AP-6	Apuesta activa y resultado perdedor	Apuesta perdida y saldo actualizado	Registrar la pérdida de la apuesta y ajustar el saldo.
empatar()	AP-7	Apuesta activa y resultado empate	Apuesta devuelta o ajustada	Tratar el empate: devolver la apuesta o aplicar regla de la casa.
getDineroTotal() const	AP-8	Objeto Apuesta inicializado	Retorna el dinero total disponible	Devolver el saldo/dinero total del jugador asociado.
getApuestaActual() const	AP-9	Apuesta definida	Retorna el valor de la apuesta actual	Retornar el monto que el jugador tiene apostado actualmente.
Carta(int pValorNominal, const std::string& pPalo, const Color&	C-1	Valores y palo válidos	Instancia Carta creada con	Crear carta con valor nominal,

pColor)			atributos	palo y color.
convertirAsAUno()	C-2	Carta es un As y está siendo considerada como 11	Valor del As forzado a 1	Convertir el valor del as de 11 a 1 para cálculo de puntaje.
getValor() const	C-3	Carta inicializada	Retorna el valor numérico	Devolver el valor nominal de la carta para cálculo de puntaje.
getPalo() const	C-4	Carta inicializada	Retorna el palo	Devolver el palo de la carta (tréboles, corazones, etc.).
getColor() const	C-5	Carta inicializada	Retorna el color enumerado	Devolver el color (enum) asociado a la carta.
Baraja()	B-1	Ninguna baraja en memoria o crear nueva	Objeto Baraja creado con estructura vacía o poblada	Inicializar la baraja, normalmente llamando a crearBaraja().
crearBaraja()	B-2	Baraja no poblada	Baraja con 52 cartas estándar generadas	Generar las 52 cartas que componen la baraja estándar.
barajar()	B-3	Baraja poblada	Orden aleatorio aplicado a la baraja	Aplicar algoritmo de barajado para mezclar las cartas.

repartirCarta()	B-4	Baraja no vacía	Se extrae y retorna la carta superior	Extraer la carta superior de la baraja y entregarla al solicitante.
estaVacía() const	B-5	Baraja inicializada	Retorna verdadero si no quedan cartas	Indicar si la baraja tiene cartas remanentes o está vacía.
Jugador()	J-1	Datos del jugador disponibles	Jugador instanciado con mano vacía	Crear objeto Jugador con atributos iniciales y mano vacía.
resetear()	J-2	Jugador tiene mano previa	Mano limpia y estado listo para nueva ronda	Vaciar la mano y restablecer estado del jugador para nueva ronda.
quierePedir() const	J-3	Turno del jugador	Retorna booleano con decisión	Obtener la decisión del jugador (si quiere pedir carta) basada en entrada o IA.
setDecision(bool decision)	J-4	Jugador activo	Decision almacenada	Registrar la decisión del jugador sobre pedir o plantarse.
Dealer()	D-1	Inicialización del dealer	Dealer instanciado con mano vacía	Crear dealer con comportamiento automático y mano vacía.

revelarCarta()	D-2	Dealer con cartas repartidas (una oculta)	Carta oculta revelada	Descubrir la carta oculta del dealer para mostrar a jugadores.
debePedir() const	D-3	Dealer en turno de juego	Retorna booleano si debe pedir según reglas	Evaluar la regla del dealer (ej. pedir si valor < 17).
agregarCarta(const Carta& carta)	MB-1	Carta válida	Carta añadida a la colección de la mano	Agregar la carta a la mano del jugador/dealer .
resetearMano()	MB-2	Mano con cartas previas	Mano vacía	Vaciar la mano para comenzar nueva ronda.
ajustarMano()	MB-3	Mano con ases u otras condiciones	Mano ajustada para mejor puntaje	Ajustar valores de ases u otras reglas para optimizar el puntaje.
getTipoDeMano() const	MB-4	ManoBase implementada	Retorna tipo de mano concreto (virtual)	Retornar el tipo de mano concreto (implementación en subclase).
getValorDeMano() const	MB-5	Mano con cartas	Retorna valor calculado	Devolver el puntaje actual de la mano.
getConteoDeCartas() const	MB-6	Mano inicializada	Retorna la cantidad de cartas	Devolver el número de cartas presentes en la mano.

getMano() const	MB-7	Mano inicializada	Retorna referencia a vector de cartas	Devolver la colección de cartas de la mano para inspección.
Vista()	V-1	Sistema de salida disponible	Vista inicializada	Crear la instancia de la vista para interacción por consola.
limpiarPantalla() const	V-2	Interfaz ya mostrando contenido	Pantalla limpiada	Limpiar la consola/terminal para nueva presentación.
mostrarTitulo() const	V-3	Vista inicializada	Título mostrado	Mostrar el título del juego en la interfaz.
mostrarMenu() const	V-4	Vista inicializada	Menú principal mostrado	Presentar opciones del menú al usuario.
mostrarSubMenu() const	V-5	Submenú disponible	Submenú mostrado	Presentar submenú con opciones adicionales.
mostrarTexto(const std::string& texto) const	V-6	Texto preparado	Texto mostrado	Imprimir texto informativo en la consola.
añadirCartaACola(Mano mano, const std::string& valor, const std::string& palo, Color colorEnum)	V-7	Parámetros válidos	Carta preparada en cola para impresión	Agregar la representación de carta a la cola correspondiente para impresión ordenada.

<code>imprimirCola(const std::vector&lt;std::string&gt;&amp; cola) const</code>	V-8	Cola con elementos	Cola impresa en consola	Imprimir los elementos de la cola en orden.
<code>esColaVacia(std::vector&lt;std::str ing&gt; (&amp;cola)[3]) const</code>	V-9	Estructuras de cola definidas	Retorna true si todas las colas vacías	Verificar si las colas están vacías.
<code>limpiarColas()</code>	V-10	Colas definidas	Colas vaciadas	Limpiar todas las colas de impresión.
<code>imprimirMano(Mano mano)</code>	V-11	Mano válida	Mano mostrada en pantalla	Formatear e imprimir la mano con sus cartas y puntaje.
<code>imprimirCola(const std::vector&lt;std::string&gt;&amp; cola) const</code>	U-1	Cola con elementos	Imprime la cola (nota: método duplicado en Vista)	Función auxiliar que imprime una cola de strings en consola.
<code>main()</code>	M-1	Dependenci as del proyecto disponibles	Aplicación iniciada o mensaje de error	Inicializar componentes principales y arrancar el flujo del juego via Controlador.