**Instituto Nacional de Aprendizaje**

**Programador de Aplicaciones Informáticas**

**Módulo Programación Estructurada**

**Proyecto Individual**

**Estudiante: Mariel Daniela Rojas Sanchez**

**Cedula: 208030487**

**Profesor: Luis Alonso Bogantes**

**18 noviembre 2022**

Índice

[Introducción: 3](#_Toc119868684)

[Objetivo General: 4](#_Toc119868685)

[Objetivos Específicos: 4](#_Toc119868686)

[Problemas 5](#_Toc119868687)

[Blackjack: 5](#_Toc119868688)

[Dominó: 6](#_Toc119868689)

[Blackjack 7](#_Toc119868690)

[Planteamiento del Problema: 7](#_Toc119868691)

[Identificación del Problema: 7](#_Toc119868692)

[Datos de entrada: 7](#_Toc119868693)

[Datos de salida: 7](#_Toc119868694)

[Observaciones (Limitaciones / Excepciones / Suposiciones): 7](#_Toc119868695)

[Planteamiento de alternativas: 8](#_Toc119868696)

[Alternativa 1: 8](#_Toc119868697)

[Alternativa 2: 9](#_Toc119868698)

[Elección de alternativa: 10](#_Toc119868699)

[Dominó 11](#_Toc119868700)

[Planteamiento del Problema: 11](#_Toc119868701)

[Identificación del Problema: 11](#_Toc119868702)

[Datos de entrada: 11](#_Toc119868703)

[Datos de salida: 11](#_Toc119868704)

[Observaciones (Limitaciones / Excepciones / Suposiciones): 11](#_Toc119868705)

[Planteamiento de alternativas: 12](#_Toc119868706)

[Alternativa 1: 12](#_Toc119868707)

[Alternativa 2: 13](#_Toc119868708)

[Elección de alternativa: 15](#_Toc119868709)

[Conclusiones: 15](#_Toc119868710)

[Bibliografía: 16](#_Toc119868711)

# Introducción:

Este proyecto tiene como fin poner en practica todos los temas vistos a lo largo del módulo de programación estructurada; En este módulo se repasa un poco el razonamiento lógico, la metodología de soluciones de problemas y la abstracción de diferentes problemas mediante la implementación de un juego educativo.

En este proyecto se van a analizar y desarrollar dos ejercicios diferentes, el primer ejercicio se basa en un juego de naipe, llamado Blackjack o conocido popularmente como “21”, este es un juego muy común y sencillo, el objetivo del juego es lograr obtener 21 puntos o lo más cercano posible, pero sin pasarse, para lo cual cada carta cuenta con su valor especifico y a cada jugador, en este caso hay tres personas jugadoras a las cuales se le reparten una cantidad de cartas a como va avanzando el juego, y la persona que llegue a los 21 puntos o que más se aproxime entonces es el ganador.

El segundo ejercicio por resolver se trata del juego “Domino”, en el cual van a haber cuatro personas jugadoras, a las cuales se reparten 7 fichas al azar, este juego cuenta con 28 fichas, por lo que las fichas se reparten todas entre los cuatro jugadores, y con base a unas ciertas reglas de juego, el juego se va desarrollando hasta que algún jugador se quede sin fichas, que seria el ganador.

Para cada uno de los ejercicios, se va a ejecutar un planteamiento de problema, se va a identificar el problema estableciendo sus datos de entrada, salida, proceso, así como también se toman en cuenta diferentes observaciones, validaciones y limitaciones, seguidamente se hace un planteamiento de dos distintas alternativas, para la elección de la alternativa óptima, por último, se diseña un algoritmo que aporte una solución adecuada a los ejercicios planteados.

El uso de los conocimientos aprendidos a lo largo del módulo es de suma importancia, para tener una buena base en la formación como estudiantes, fomentando distintos temas y cualidades características de un buen programador, como por ejemplo aprender a aplicar la lógico-matemática, abstracción e interpretación de distintos problemas, lógica estructurada y por último a la lógica computacional.

# Objetivo General:

Programar los dos ejercicios planteados, los cuales cuenten con los requerimientos planteados en el caso dado, utilizando los principales elementos semánticos y sintácticos del modelo de programación estructurada.

# Objetivos Específicos:

Determinar las posibles soluciones a los dos ejercicios correspondientes, mediante la resolución del método de solución de problemas.

Aprender a aplicar la programación estructurada.

Crear un algoritmo eficaz y funcional, que sea capaz de poder jugar Blackjack de la manera solicitada.

Implementar un algoritmo de forma correcta, que sea funcional y eficaz para el juego “Domino”.

# Problemas

## Blackjack:

El juego de cartas conocido como Blackjack o 21 consiste en un sencillo juego cuyo objetivo es lograr obtener 21 puntos o lo más cercano posible, pero sin pasarse.

En la baraja cada carta numérica vale sus respectivos puntos, cada carta con dibujo (J, Q, K) valen 10 puntos y el As vale 1 u 11 puntos, el valor del As lo decide el jugador a su conveniencia.

Se reparten 2 cartas a cada jugador, y cada jugador decide si pide más cartas o se queda con su valor actual. Gana la persona que logre obtener 21 puntos, o lo más cercano posible, pero sin pasarse.

Realice una aplicación de consola que permita a 3 personas jugar una partida de Blackjack, no se utilizarán las reglas oficiales (dividir, doblar, rendirse, seguro, cobrar 1 a 1, etc.). Tampoco existirán las reglas de verse obligado a pedir si se tiene mayor o menor cantidad de ciertos puntos. No existirá la figura oficial del crupier (repartidor), ya que este tiene reglas diferentes.

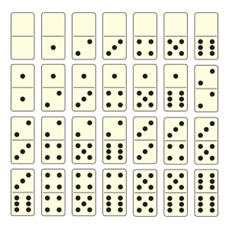
Se realizará una partida sencilla en donde al iniciar se le reparten 2 cartas a cada una de las 3 personas. Cada persona puede decidir si quiere una carta más en cada ronda, pero si se pasa de 21 automáticamente pierde.

Si una persona tiene un As se tomará como 11. Pero cuando una persona se pase de 21 automáticamente se tomará el As como 1.

En cada ronda cada persona decide si desea tomar otra carta o no, y además cada vez que una persona toma una carta se comprueba si tiene un 21 para declarar que ganó.

Cuando las 3 personas ya hayan indicado que no desean tomar más cartas, entonces el programa calculará quién ganó, tomando como ganadora a la persona cuyos puntos estén más cerca de 21, pero sin pasarse, como se indicó anteriormente. Podría haber empates.

## Dominó:

Crear una aplicación para el juego del dominó. El objetivo de un jugador es ganar la partida conforme criterios básicos de juego.

Realice una aplicación de consola utilizando el lenguaje de programación C++, que permita a 4 personas jugar una partida de dominó, no se utilizarán las reglas oficiales. Se propone una versión simplificada del juego. Se realizará una partida sencilla en donde las reglas básicas que regulan el juego del dominó se describen a continuación:

* El juego se desarrolla con cuatro jugadores, a los cuales hay que solicitarles el nombre.
* Las 28 fichas de dominó serán repartidas al azar (7 fichas a cada jugador).
* El jugador/a que inicia la partida será quien tenga la ficha 6/6 (doble seis), continuando el turno por el siguiente jugador según el orden de ingreso. En su turno, cada jugador colocará una de sus piezas con la restricción de que dos piezas solo pueden colocarse juntas cuando los cuadrados adyacentes son del mismo valor. Si un jugador no puede colocar ninguna ficha en su turno, tendrá que pasar el turno al siguiente jugador.
* El final de la partida puede venir determinado por dos situaciones: (1) un jugador colocó la última de sus fichas y, por tanto, ganó la partida; o (2) todos los jugadores tienen aún fichas, pero ninguno puede colocar ninguna de ellas. En este último caso, ganará la partida el jugador cuyas fichas sumen menos puntos.

# Blackjack

## Planteamiento del Problema:

Desarrollar el juego “Blackjack”’, el cual se basa en un juego de naipe, donde la persona que logre acercarse o obtener 21 puntos es la persona ganadora.

## Identificación del Problema:

### Datos de entrada:

* Nombre de los tres jugadores a participar.
* Número para empezar a jugar, para seguir jugando o para salir.
* Letra para solicitar cartas en cada ronda.

### Datos de salida:

* Mensaje indicando las instrucciones.
* Mensaje indicando cual jugador comienza.
* Mensaje en cada ronda preguntando si desean una carta o no.
* Interfaz para visual los puntos de cada jugador.
* Mensaje imprimiendo el resultado final.
* Mensaje preguntando si desea continuar jugando o desea salir.

## Reglas de negocio:

* No se utilizarán las reglas oficiales (dividir, doblar, rendirse, seguro, cobrar 1 a 1, etc.).
* No existirán las reglas de verse obligado a pedir si se tiene mayor o menor cantidad de ciertos puntos.
* No existirá la figura oficial del crupier (repartidor), ya que este tiene reglas diferentes.

## Observaciones (Limitaciones / Excepciones / Suposiciones):

* La baraja de naipe contiene 52 cartas, 13 cartas de 1 al As de Corazones, 13 cartas del a al As de Diamantes, 13 cartas de 1 al As de Trebols, 13 cartas de 1 al As de Picas.
* En la baraja cada carta numérica vale sus respectivos puntos, cada carta con dibujo (J, Q, K) valen 10 puntos y el As vale 1 u 11 puntos, el valor del As lo decide el jugador a su conveniencia.
* Si una persona tiene un As se tomará como 11. Pero cuando una persona se pase de 21 automáticamente se tomará el As como 1.
* En cada ronda cada persona decide si desea tomar otra carta o no.
* Cada vez que una persona toma una carta se comprueba si tiene un 21 o más de 21 para declarar que ganó o para indicarle que ya perdió.
* Cuando las 3 personas ya hayan indicado que no desean tomar más cartas, entonces el programa calculará quién ganó.
* Gana la persona cuyos puntos estén más cerca de 21, pero sin pasarse.
* Podría haber empates.
* Los jugadores deciden si quieren seguir jugando o quieren terminar la partida.

## Planteamiento de alternativas:

### Alternativa 1:

1. Inicio.
2. Se incluyen las librerías necesarias a utilizar.
3. Se declaran todas las variables necesarias.
4. Se le da la bienvenida al juego.
5. Se imprimen en pantalla algunas indicaciones sobre el juego.
6. Se le solicitan al usuario sus nombres.
7. Se le solicita que ingrese un digito para comenzar el juego o si desea salir.
8. Si el usuario desea jugar, el juego comienza, si no, el programa termina inmediatamente.
9. Se crea una estructura de cartas, la cual va a estar compuesta por sus símbolos respectivos y sus números.
10. Se crea una estructura de jugadores el cual está compuesto de cartas.
11. Se crea la baraja de naipe, con un ciclo el cual vaya recorriendo un vector el cual es de 52 espacios.
12. La baraja de naipe se mezcla.
13. Se reparte la baraja de naipe, se reparten 2 cartas a cada uno de los tres jugadores.
14. Se crea un turno con la función rand para ver que jugador comienza.
15. Se imprime en pantalla cual jugador comienza.
16. Se les pregunta a los jugadores en cada ronda si desea recibir una carta o no, esto se repite hasta que todos los jugadores digiten que no, o hasta que algún jugador llegue a 21, o hasta que todos los jugadores se pasen de 21.
17. Dentro del ciclo descrito en el punto 16, se hacen validaciones, con sumas y con estructuras de decisión donde se va validando los puntos de los tres jugadores, para llevar la cuenta y el control, con el fin de verificar cual jugador se pasa de 21 puntos, llega a los 21 puntos exactos y gana, o cual jugador se aproxima más.
18. Se imprime en pantalla el resultado final, ya sea si hubo un jugador ganador, o hubo empate.
19. Se pregunta si desean jugar una nueva partida o si desean finalizar el programa.
20. Fin.

### Alternativa 2:

1. Inicio.
2. Se incluyen las librerías necesarias a utilizar.
3. Se declaran todas las variables necesarias.
4. Se le da la bienvenida al juego.
5. Se imprimen en pantalla algunas indicaciones sobre el juego.
6. Se le solicitan al usuario sus nombres.
7. Se crea una estructura de cartas, la cual va a estar compuesta por sus símbolos respectivos y sus números.
8. Se crea una estructura de jugadores el cual está compuesto de cartas.
9. Se le solicita que ingrese un digito para comenzar el juego o si desea salir.
10. Si el usuario desea jugar, el juego comienza, si no, el programa termina inmediatamente.
11. En una función aparte del programa principal crea la baraja de naipe, con un ciclo el cual vaya recorriendo un vector el cual es de 52 espacios, esta función se llama en el programa principal, dentro del ciclo donde comienza el juego hasta que el juego termina.
12. En una función aparte, se mezcla la baraja de naipe y esta función se llama en el programa principal.
13. Se crea una función que se encarga de repartir la baraja de naipe, se reparten 2 cartas a cada uno de los tres jugadores, y las cartas que se van repartiendo se van descontando de la baraja, esta función se llama en el programa principal.
14. En el programa principal se crea un turno con la función rand para ver que jugador comienza.
15. Se imprime en pantalla cual jugador comienza.
16. En el programa principal, dentro de un ciclo, se les pregunta a los jugadores en cada ronda si desea recibir una carta o no, esto se repite hasta que todos los jugadores digiten que no, o hasta que todos los jugadores se pasen de 21 o bien, hasta que algún jugador alcance los 21 puntos.
17. Dentro del ciclo descrito en el punto 16, se hacen validaciones, con sumas y con estructuras de decisión donde se va validando los puntos de los tres jugadores, para llevar la cuenta y el control, con el fin de verificar cual jugador se pasa de 21 puntos, llega a los 21 puntos exactos y gana, o cual jugador se aproxima más.
18. Se imprime en pantalla el resultado final, ya sea si hubo un jugador ganador, o hubo empate.
19. Se pregunta si desean jugar una nueva partida o si desean finalizar el programa.
20. Fin.

## Elección de alternativa:

De mi punto de vista, considero que la alternativa 2 es mucho más eficaz y funcional, su algoritmo es más sencillo de entender, más ordenado, se facilita el buscar errores y de modificarlos, ya que utiliza diferentes funciones, utilizando así principio de la funcionalidad.

**Dominó**

Planteamiento del Problema:

Crear una aplicación para el juego del dominó el cual se basa en 28 fichas y cuenta con cuatro jugadores. El objetivo de un jugador es ganar la partida.

Identificación del Problema:

Datos de entrada:

* Nombre de los tres jugadores a participar.
* Número para empezar a jugar, para seguir jugando o para salir.

Datos de salida:

* Mensaje indicando las instrucciones.
* Mensaje indicando cual jugador comienza.
* Mensaje indicando los turnos de cada jugador.
* Interfaz para visualizar el campo de juego.
* Mensaje imprimiendo el resultado final.
* Mensaje preguntando si desea continuar jugando o desea salir.

## Reglas de negocio:

* No se utilizarán las reglas oficiales. Se propone una versión simplificada del juego.

## Observaciones (Limitaciones / Excepciones / Suposiciones):

* Únicamente pueden jugar 4 jugadores, ni más ni menos.
* Son 28 fichas las cuales se reparten 7 a cada jugador, por lo que no sobran fichas.
* Cada ficha está dividida en 2 espacios iguales en los que aparece una cifra de 0 hasta 6. Las fichas cubren todas las combinaciones posibles con estos números.
* Inicia la partida el jugador que tenga la ficha 6/6 (doble seis).
* No habrá tiempo de juego. Cada jugador se toma su tiempo necesario para jugar
* Después del primer jugador, los turnos se dan en orden continuando el turno por el siguiente jugador según el orden de ingreso.
* Únicamente se puede colocar una ficha por turno.
* Dos piezas solo pueden colocarse juntas cuando los cuadrados adyacentes son del mismo valor.
* Si un jugador no puede colocar ninguna ficha en su turno, tendrá que pasar el turno al siguiente jugador.
* El jugador que coloque sus 7 fichas es el ganador.
* Si ningún jugador puede color sus fichas, el juego termina, y el ganador será el jugador cuyas fichas sumen la menor cantidad de puntos.
* Es poco probable que existan empates, pero de igual manera se validan.

Planteamiento de alternativas:

Alternativa 1:

1. Inicio.
2. Se incluyen las librerías necesarias a utilizar.
3. Se declaran todas las variables necesarias.
4. Se le da la bienvenida al juego.
5. Se imprimen en pantalla algunas indicaciones sobre el juego.
6. Se le solicitan al usuario sus nombres.
7. Se le solicita que ingrese un digito para comenzar el juego o si desea salir.
8. Si el usuario desea jugar, el juego comienza, si no, el programa termina inmediatamente.
9. Se crea una estructura de fichas, son 28 fichas, las cuales van está dividida en 2 espacios iguales en los que aparece una cifra de 0 hasta 6. Las fichas cubren todas las combinaciones posibles con estos números.
10. Se crea una estructura de jugadores el cual está compuesto de dichas fichas.
11. Las 28 fichas se mezclan.
12. Se reparte la baraja de naipe, se reparten 7 fichas a cada uno de los cuatro jugadores.
13. El jugar que tenga la ficha 6/6 es el jugador que comienza.
14. Se imprime en pantalla cual jugador comienza.
15. El segundo turno sigue después del primer jugador, los turnos se dan en orden continuando el turno por el siguiente jugador según el orden de ingreso.
16. Se le pregunta a los jugadores si desea colocar una ficha o si desea brincarse el turno, si el jugador desea colocar una ficha, esta ficha debe validarse, que alguno de sus lados sea igual a alguno de los lados de las ultimas fichas del campo de juego, si la ficha que se desea colocar no es valida o permitida, entonces se le indica al jugador que no es posible colocarla, que intente con otra, pero si la ficha si es valida entonces se le descuenta una ficha a su total de fichas, para llevar el control de cuantas fichas tiene cada jugador.
17. Si ningún jugador puede colocar fichas validas, entonces el juego termina y el ganador será el jugador que cuyas fichas sumen la menor cantidad de puntos, si no, cuando las fichas de cada jugador sean igual a cero, entonces ese jugador es el ganador.
18. Se imprime en pantalla el resultado final, ya sea si hubo un jugador ganador, o hubo empate (Casi imposible un empate).
19. Se pregunta si desean jugar una nueva partida o si desean finalizar el programa.
20. Fin.

Alternativa 2:

1. Inicio.
2. Se incluyen las librerías necesarias a utilizar.
3. Se declaran todas las variables necesarias.
4. Se le da la bienvenida al juego.
5. Se imprimen en pantalla algunas indicaciones sobre el juego.
6. Se le solicitan al usuario sus nombres.
7. Se crea una estructura de fichas, son 28 fichas, las cuales van está dividida en 2 espacios iguales en los que aparece una cifra de 0 hasta 6. Las fichas cubren todas las combinaciones posibles con estos números.
8. Se crea una estructura de jugadores el cual está compuesto de dichas fichas.
9. Se le solicita que ingrese un digito para comenzar el juego o si desea salir.
10. Si el usuario desea jugar, el juego comienza, si no, el programa termina inmediatamente, para esto es necesario una estructura de iteración, donde todos los puntos siguientes, hasta el fin van a estar o ser llamados dentro de este ciclo.
11. En una función aparte del programa principal se hace un ciclo, donde las 28 fichas se van a mezclar, esta función se llama en el programa principal.
12. Dentro del programa principal se reparten las 7 fichas a cada uno de los cuatro jugadores.
13. El jugar que tenga la ficha 6/6 es el jugador que comienza.
14. Se imprime en pantalla cual jugador comienza.
15. El segundo turno sigue después del primer jugador, los turnos se dan en orden continuando el turno por el siguiente jugador según el orden de ingreso.
16. En una función aparte del programa principal, se le pregunta a los jugadores si desea colocar una ficha o si desea brincarse el turno, si el jugador desea colocar una ficha, esta ficha debe validarse, que alguno de sus lados sea igual a alguno de los lados de las ultimas fichas del campo de juego, si la ficha que se desea colocar no es válida o permitida, entonces se le indica al jugador que no es posible colocarla, que intente con otra, si ninguna ficha es válida, entonces se pasa el turno al siguiente jugador, pero si la ficha si es válida entonces se le descuenta una ficha a su total de fichas, para llevar el control de cuantas fichas tiene cada jugador,
17. Si ningún jugador puede colocar fichas validas, entonces el juego termina y el ganador será el jugador que cuyas fichas sumen la menor cantidad de puntos, si no, cuando las fichas de cada jugador sean igual a cero, entonces ese jugador es el ganador.
18. Se imprime en pantalla el resultado final, ya sea si hubo un jugador ganador, o hubo empate (Casi imposible un empate).
19. Se pregunta si desean jugar una nueva partida o si desean finalizar el programa.
20. Fin.

Elección de alternativa:

En mi opinión considero que la alternativa 2 es más eficaz y funcional, su algoritmo es más sencillo de entender, más claro, ordenado, se facilita el buscar errores y de modificarlos, ya que utiliza diferentes funciones, utilizando así principio de la funcionalidad, lo que hace que el tiempo de desarrollo sea menor y que la calidad del proyecto sea la más adecuada.

# Conclusiones:

En este proyecto del módulo de programación estructurada se han puesto en práctica básicamente todos los temas estudiados a lo largo del curso, como lo son: diferentes estructuras de decisión, estructuras cíclicas o de iteración, sentencias de salto, uso de variables constantes, el uso de funciones, de diferentes librerías y bibliotecas, soluciones a diferentes errores de sintaxis, errores de lógica o ejecución, así como también la utilización de diferentes operadores

Para la realización de este proyecto tuve que investigar bastante diferentes soluciones en cada uno de los pasos de las alternativas propuestas, como por ejemplo para validar algunas opciones de juego; También fue necesario investigar algunas librerías y bibliotecas para implementar en el proyecto.

Las alternativas de solución son de suma importancia para tener una mayor claridad de cómo resolver los ejercicios propuestos y tener dos formas diferentes de resolver un mismo problema, para llegar a la solución más adecuada.

La ejecución de estos ejercicios que se basan en juegos de mesa, es de suma importancia para implementar la lógica, ya que se requiere hacer un pensamiento matemático y lógico para que todo funcione de la mejor manera, con respecto a todas las observaciones, suposiciones y las reglas de negocio.

# Bibliografía:

<https://cplusplus.com/>

<https://www.mundijuegos.com.mx/multijugador/domino/reglas/>