Reporte de proyecto final equipo 1

**Nombre del proyecto:** Proyecto de ejemplo 1

**Descripción del proyecto:** Añadir un párrafo de máximo 100 palabras para describir el proyecto y los resultados obtenidos

**Integrantes del equipo:**

* Carmona de Jesus David 1, 202370355, ingeniería en ciencias de la computación, **calificación:**
* Israel Trejo Hernández 2, 202378748, carrera ingeniería en ciencias de la computación, **calificación:**
* José Emiliano Cabrera Villarreal 3, 202370208, Ingeniería en ciencias de la computación, **calificación:**
* Joshua Anwa Rodríguez Castellano 4, 202365752, carrera ingeniería en ciencias de la computación, **calificación:**
* Ricardo Xochitla Guzmán 5, 202368692, carrera ingeniería en ciencias de la computación, **calificación:**

**Profesor:** Dr. Carlos Leopoldo Carreón Díaz de León

**Fecha de entrega:** ***29 de noviembre del 2023***

# Introducción

Introducción

La búsqueda constante de métodos innovadores para mejorar la salud y el bienestar ha llevado al desarrollo de un proyecto revolucionario: un algoritmo avanzado en C++ destinado a la generación personalizada de rutinas de ejercicio. Este proyecto no solo se distingue por su eficiencia y versatilidad, sino también por su enfoque inclusivo, brindando rutinas adaptadas a las necesidades específicas de individuos con diversos desafíos de salud. A lo largo de estas dos hojas, exploraremos los fundamentos, la funcionalidad y el impacto potencial de este algoritmo, destacando sus capacidades para abordar las limitaciones físicas y médicas de una amplia gama de usuarios.

Fundamentos del Proyecto

En el corazón de este proyecto se encuentra un algoritmo de generación de rutinas basado en el lenguaje de programación C++. Este lenguaje, conocido por su eficiencia y versatilidad, proporciona la plataforma perfecta para la implementación de un sistema que requiere un procesamiento rápido y preciso de datos. El enfoque principal del algoritmo es utilizar funciones esenciales para crear rutinas de ejercicio personalizadas, adaptadas a los objetivos, niveles de condición física y preferencias individuales de los usuarios.

Adaptabilidad y Personalización

Una de las características sobresalientes de este algoritmo es su capacidad para adaptarse a una amplia variedad de usuarios, independientemente de su condición física o limitaciones médicas. El sistema no solo tiene en cuenta los objetivos de acondicionamiento físico generales, sino que también aborda las necesidades específicas de aquellos con problemas de respiración, limitaciones de movilidad, afecciones cardíacas o lesiones. La inclusión de parámetros específicos y la capacidad de ajuste continuo hacen que este algoritmo sea excepcionalmente versátil, ofreciendo soluciones de ejercicio personalizadas para cada individuo.

Rutinas Especiales para Desafíos de Salud

El proyecto va más allá de la generación convencional de rutinas de ejercicio al incorporar rutinas especializadas diseñadas para abordar desafíos de salud específicos. Para aquellos con problemas respiratorios, el algoritmo adapta la intensidad y el tipo de ejercicio para mejorar la capacidad pulmonar de manera gradual y segura. En el caso de personas con limitaciones de movilidad, se implementan ejercicios que fortalecen áreas específicas, fomentando la mejora progresiva. Para aquellos con condiciones cardíacas, el algoritmo ajusta la intensidad y la duración del ejercicio para garantizar la seguridad cardiovascular. Además, para personas con lesiones, se generan rutinas que evitan la sobrecarga en áreas afectadas, favoreciendo la rehabilitación.

Interfaz de Usuario Intuitiva y Accesible

La implementación de este algoritmo se acompaña de una interfaz de usuario intuitiva y accesible. Esto garantiza que los usuarios, independientemente de su experiencia en el ámbito tecnológico, puedan acceder fácilmente a las rutinas generadas. La interfaz permite la personalización adicional, permitiendo a los usuarios ajustar sus preferencias y objetivos en cualquier momento. Además, se ha prestado especial atención a la presentación visual de las rutinas, facilitando su comprensión y seguimiento.

En resumen, este proyecto representa un avance significativo en la optimización de la salud y el bienestar a través de la tecnología. Al proporcionar un algoritmo en C++ que no solo genera rutinas de ejercicio personalizadas, sino que también aborda desafíos de salud específicos, se sientan las bases para un enfoque inclusivo en la promoción de la actividad física. En las siguientes secciones, exploraremos en detalle las funciones clave del algoritmo, su implementación práctica y el impacto potencial en la mejora de la calidad de vida de los usuarios.

# Descripción del proyecto

En esta sección se reporta que es lo que debe hacer el código desarrollado para el proyecto, como se han solucionado los problemas presentados, y hasta donde fue posible realizar el proyecto.

Se debe incluir el código del proyecto utilizando la siguiente notación:

El proyecto presenta un algoritmo en C++ que genera rutinas de ejercicio personalizadas y aborda desafíos de salud específicos. El código se enfoca en la adaptabilidad, permitiendo a los usuarios definir objetivos y preferencias para generar rutinas dinámicas. Se ha solucionado la adaptación continua de las rutinas, asegurando una progresión segura y eficaz. Para usuarios con problemas de salud, el algoritmo genera rutinas especializadas, abordando problemas respiratorios, limitaciones de movilidad, afecciones cardíacas o lesiones. Se ha optimizado la eficiencia del código y la interfaz del usuario para una experiencia fluida. El proyecto representa una solución integral para mejorar la salud y el bienestar a través de la tecnología.

Tipo de letra: Courier Std

|  |
| --- |
| #include <iostream>  #include <stdio.h>  #include <stdlib.h> //Sirve para convertir un entero a cadena de caracteres.  #include <cctype> //Convierte a mayúsculas un carácter o un intervalo de caracteres  #include <cstdlib>// Sirve para ejecutar subprocesos o comandos del sistema operativo  #include <fstream> //para los archivos  #include <vector> //sirve para crear y usar vectores  #include <string.h> //para utlizar strings.  #include "User.h"  using namespace std;  vector <User> users; //Es un vector que contiene objetos de tipo User  vector<string> dia\_semana;  //vector <string> ejercicios={"Domingo","Lunes","Martes","Miercoles","Jueves","Viernes","Sabado"}; vector que dice que dia es, solo se ocuapara en caso de que lo vemaos necesario |

|  |
| --- |
| //Funcion para guardar los usuarios resgistrados en un archivo. tiene 36 lienas de código.  void saveUsersToFile(std::vector<User> use, const char\* filename)  {  std::ofstream outfile(filename );  if (outfile.is\_open())  {  int n = use.size();  outfile.write((char\*)&n, sizeof(int));  for (int i=0; i<n;i++)  {  int id= users[i].getID();  std::string name=users[i].getName();  std::vector<char> bytes(name.begin(), name.end());  std::string password=users[i].getPassword();  std::vector<char> byt(password.begin(), password.end());  char lesiones=users[i].getLesiones();  char enfermedades=users[i].getEnfermedades();  float altura=users[i].getAltura();  float peso=users[i].getPeso();  int edad=users[i].getEdad();  char sexo=users[i].getSexo();  int tamaniovector=bytes.size();  int tamaniopass=byt.size();  outfile.write((char\*)&id, sizeof(int));  outfile.write((char\*)&tamaniovector, sizeof(int));  outfile.write(bytes.data(), bytes.size());  outfile.write((char\*)&tamaniopass, sizeof(int));  outfile.write(byt.data(), byt.size());  outfile.write((char\*)&altura, sizeof(float));  outfile.write((char\*)&lesiones, sizeof(char));  outfile.write((char\*)&peso, sizeof(float));  outfile.write((char\*)&enfermedades, sizeof(char));  outfile.write((char\*)&sexo, sizeof(char));  outfile.write((char\*)&edad, sizeof(int));  }  outfile.close();  }  } |

|  |
| --- |
| // Funcion que lee un archivo que contiene los usuarios, y los almacena en un vector, ese vector almacena objetos de tipo User, con sus respectivos parametros.  //contiene 30 lineas de codigo  void readUsersFromFile(const char\* filename)  {  vector<User> usuarios;  ifstream infile(filename );  int n;  infile.read((char\*)&n, sizeof(int));  for (int i = 0; i <n; i++)  {  int edad,id, vectortam, tamaniopass;  float altura, peso, imc;  char sexo, enfermedades, lesiones;  infile.read((char\*)&id, sizeof(int));  infile.read((char\*)&vectortam, sizeof(int));  std::vector<char> bytes(vectortam);  infile.read(bytes.data(), bytes.size());  infile.read((char\*)&tamaniopass, sizeof(int));  std::vector<char> bytpass(tamaniopass);  infile.read(bytpass.data(), bytpass.size());  infile.read((char\*)&altura, sizeof(float));  infile.read((char\*)&lesiones, sizeof(char));  infile.read((char\*)&peso, sizeof(float));  infile.read((char\*)&enfermedades, sizeof(char));  infile.read((char\*)&sexo, sizeof(char));  infile.read((char\*)&edad, sizeof(int));  std::string name(bytes.begin(), bytes.end());  std::string password(bytpass.begin(), bytpass.end());  User user(name, id, password, altura, lesiones, peso, enfermedades, edad, sexo);  users.push\_back(user);  }  infile.close();  } |

|  |
| --- |
| //Funcion que recibe como parametros un string, que fue ingresado previamente por el usuario, el cual va a validar como entrada unicamente letras  // en caso de que no sean letras le indicara al usuario que introduzca solo letras, y estara en un bucle hasta que lo haga correctamente.  // 26 lineas de codigo  string OnlyLetters(string tipovalor) //Validacion de entrada de solo letras.  {  bool flagss;  int conta, lm=0;  string name;  do{ //recorre todo el string, analizando si solo introdujo letras, en caso de que no el ciclo se repetira hasta que lo realice correctamente  system("cls");  conta=0;  cout<<tipovalor<<" de usuario: "<<endl;  getline(cin, name);  for(int i=0; i<name.size();i++)  {  lm=(int)name[i];  if(((lm<65||lm>90) && (lm<97 || lm>122) ) && lm!=32 && (lm<160 || lm>165) )  conta++;  }  if(conta>0)  {  cout<<"Introduzca solo letras"<<endl;  system("pause");  system("cls");  flagss=true;  }  else flagss=false;  }while(flagss||name.empty());//el ciclo se repetira hasta que el usuario solo introduja letras, name.empty se expresa en caso de que no halla leido nada el string,  return name;  } |

|  |
| --- |
| //Esta funcion es similar a la anterior solo que este caso solo validara numeros  // 38 lineas de codigo  float OnlyNums(string tipodato)//validacion de entrada de solo numeros  {  bool flagss=true;  int conta, sd=0;  float valor;  string numeros;  do{  system("cls");  conta=0;  cout<<tipodato<<" de usuario: "<<endl;  getline(cin, numeros);  for(int i=0; i<numeros.size();i++)  {  sd=(int)numeros[i];  if( (sd<48 || sd>57) && sd!=46 )  conta++;  }  if (conta>0)  {  cout<<"Introduzca solo numeros"<<endl;  system("pause");  flagss=true;  }  else flagss=false;  }while(flagss||numeros.empty());  valor= std::stof(numeros);  return valor;  } |

|  |
| --- |
| //Solo muestra en pantalla los datos del usuario, recibe como parametro un objeto User  //12 lineas de codigo  void MostrarDatos(User u)  {  std::cout<<"Tu informacion es la siguiente: "<<std::endl;  cout<<"Nombre: "<< u.getName() <<endl;  cout<<"Edad: "<<u.getEdad() <<endl;  cout<<"Peso: "<<u.getPeso()<<endl;  cout<<"Altura: "<<u.getAltura()<<endl;  cout<<"Sexo: "<<u.getSexo()<<endl;  cout<<"Enfermedades: "<<TipoDeEnfermedad(u.getEnfermedades())<<endl;  cout<<"Lesiones: "<<TipoDeFractura(u.getLesiones())<<endl;  cout<<"IMC: "<< (u.getPeso()/(u.getAltura()\*u.getAltura()) )<<endl;  cout<<"IMC: "<< (u.getPassword() )<<endl;  } |

|  |
| --- |
| //Funcion para Determinar la eleccion de si o no, marcada por el usuario, recibe como parametro un string previamente leido, y devuelve un dato de tipo bool, para seguir operando  //23 lineas de codigo  bool SiNo(string lees)  {  bool flag, tiene;  char condiciontype;  do{  system("cls");  cout<< lees << " ? Y/N"<<endl;  cin>> condiciontype;  cin.ignore();  if (tolower(condiciontype)!= 'y' && tolower(condiciontype)!='n')  {  cout<<"ingrese una opcion valida:"<<condiciontype<<endl;  system("pause");  flag=true;  }  else{  if (tolower(condiciontype)== 'y')  tiene=true;  else{tiene=false;}  flag=false;  }  }while(flag);  return tiene;  } |

|  |
| --- |
| //Funcion que recibe un string leido, y un numero que define las opciones correctas que presenta un menu, devuelve un valor tipo booleano, dependiendo si ingreso una opcion valida o no  //este se determina con valores de la tabla ascci.  //33 lineas de codigo  bool OpcionesValidas(string opc, int ascci)  {  bool rep;  int sd,conta=0,lm;  try{  lm=std::stoi(opc);  }catch(std::invalid\_argument& e){conta=1;}  if(opc.size()==1)  {  for(int i=0; i<opc.size();i++)  {  sd=(int)opc[i];  if(sd<49 || sd>ascci)  conta++;  }  if (conta>0)  {  cout<<"ingrese una opcion valida"<<endl;  system("pause");  system("cls");  rep=true;  }  else rep=false;  }  else  {  cout<<"ingrese una opcion valida"<<endl;  system("pause");  system("cls");  rep=true;  }  return rep;  } |

|  |
| --- |
| // muestra en pantalla, un menu de opciones previamente pasado por un string, y el numerod e opciones validas, determina mediante la funcion anterior si tiene una opcionvalida o no  //en caso de que no, seguira en el do while, hasta que marque una opcion valida.  //11 lineas de codigo  char MenusDeOpciones(string opciones, int asci)  {  char opcion;  string vl;  do{  cout<<opciones<<endl;  cin>>opcion;  cin.ignore();  vl=opcion;  }while(OpcionesValidas(vl, asci));  return opcion;  } |

|  |
| --- |
| //Funcion que determina si el usuario tiene una enfermedad, llama dos funciones, la primera que valida si tiene o no una enfermedad,  //la segunda funcion que llama, determina que tipo de enfermedad tiene.  //12 lineas de ocdigo  char Enfermedad()  {  string enfermedades;  int menenfer;  char logs;  bool flags;  flags= SiNo("Usted presenta alguna enfermedad que le impida ejercitarse adecuadamente");  if(flags)  logs=MenusDeOpciones("Que tipo de enfermedad presenta?\n [1] Cardiaca \n[2] Pulmonar \n[3] Artritis \n[4] Cardiovascular ", 52);  else { logs='0' ;}  return logs;  } |

|  |
| --- |
| //Es una funcion muy similar a la anterior, pero en este caso determina si tiene o no tiene lesion, y que tipo de lesion tiene.  //10 lineas de codigo  char Lesion()  {  string lesiones;  bool flags;  char logs;  flags= SiNo("Usted presenta alguna lesión que le impida ejercitarse adecuadamente");  if(flags)  logs=MenusDeOpciones("Que tipo de lesion presenta? \n[1] Fractura \n[2] Dislocacion \n[3] Esguience \n[4] Desgarre \n[5] Tendinitis ",53);  else {logs='0';}  return logs;  } |

|  |
| --- |
| //esta funcion devuelve un string, que determina y lee del usuario la contrasenia que quiere poner el usuario, y compara para que este seguro de poner esa contrasenia  //28 lineas de codigo  string Pass()  {  string password,cpassword;  bool flag;  do{  system("cls");  cout<<"Introduce tu contraseña: "<<endl;  getline(cin, password);  if(!password.empty())  {  cout<<"Confirma tu contraseña: "<<endl;  getline(cin, cpassword);  if(password!=cpassword)  {  cout<<"Las contraseña no coinciden:"<<endl;  cout<<"Vuleve a intentarlo"<<endl;  system("pause");  system("cls");  flag=true;  }  else flag=false;  }  else flag=true;  }while(flag);  return password;  } |

|  |
| --- |
| //Esta funcion, le pide al usuario si desea hacer cambios a los datos que puso, previamente, se ejecuta cada vez que se registra el usuario  //si se llego a equivocar, y cuando inicia sesion, por si quiere cambiar algun parametro  //53 lineas de codigo  void Cambios(int id)  {  bool flag,validar;  char cambiar,enfermedad, lesion;  string name, cpassword;  float peso, altura;  int edad;  while(SiNo("Desea Realizar algun cambio? "))  {  cambiar=MenusDeOpciones("Que paranetro quieres cambar?\n[1] Nombre \n[2] Peso \n[3] Edad \n[4] Enfermedad \n[5] Lesion \n[6] Altura \n[7] Password ", 55 );  switch(cambiar)  {  case 49:  name=OnlyLetters("Nombre");  users[id].setName(name);  break;  case 50:  peso=OnlyNums("Peso");  users[id].setPeso(peso);  break;  case 51:  edad=OnlyNums("Edad");  users[id].setEdad(edad);  break;  case 52:  enfermedad=Enfermedad();  users[id].setEnfermedades(enfermedad);  break;  case 53:  lesion=Lesion();  users[id].setLesiones(lesion);  break;  case 54:  altura=OnlyNums("Altura");  users[id].setAltura(altura);  break;  case 55:  cpassword=Pass();  users[id].setPassword(cpassword);  break;  }  MostrarDatos(users[id]);  saveUsersToFile(users, "usuariosdata.txt");  system("pause");  }  } |

|  |
| --- |
| //Dependiendo que dia de la semana devolvera false o true, que servira para ver si es dia de descanso o de entrenamiento;  bool DiaSem(int day)  {  bool flag;  if(day==0 || day==2 ||day==4 || day==6)  flag=false;  else flag=true;  return flag;  } |

|  |
| --- |
| bool TipoDeEnfermedad(int id )  {  bool enfer;  if((int) users[id].getEnfermedades()==48)  enfer=true;  else enfer=false;  return enfer;  } |

|  |
| --- |
| string TipoDeFractura(int id)  {  string frac;  return frac;  } |

|  |
| --- |
| //Funcion que arma la rutina para el usuario, como parametro el indice donde se encunetra el usuario  void Gym\_day(int id)  {  //indica que dia de la semana nos encontradmos  int day;  char pref;  time\_t now=time(0);  tm \* time=localtime(&now);  day=time-> tm\_wday;  day=3;  if(DiaSem(day)){  pref=MenusDeOpciones("Como prefieres trabajar \n[1]Peso libre \n[2]Maquinas \n [3]Ambas", 51);  switch(pref)  {  case 49:  {  TipoDeEnfermedad(id);  }  break;  case 50:  {  }  break;  case 51:  {  }  break;  }  }  else{  system("cls");  cout<<"\n \n Recuerda que los dias de descanso tambien son importantes\n para que tu cuerpo se recupere, y evitemos posibles lesiones o se intensifiquen"<<endl;  system("pause");  }  } |

|  |
| --- |
| //Registro de un usuario, donde pedira sus daros  //40 lineas de codigo  void Registro\_User()  {  std:: string name, password, cpassword,sexo;  char condiciontype, sex, enfermedades, lesiones;  int id,edad;  float peso, altura;  bool flag;  cout<<"----------Bienvenido----------\n --Ingresa tus datos correctamente--"<<endl;  name=OnlyLetters("Nombre");  edad=OnlyNums("Edad");  peso=OnlyNums("Peso");  altura=OnlyNums("Altura");  do{  cout<<"Ingresa tu sexo: \n [H]Hombre \n [M]Mujer "<<endl;  cin>>sex;  cin.ignore();  if( tolower(sex)!='h' && tolower(sex)!='m')  {  cout<<"Ingrese un sexo valido"<<endl;  system("pause");  flag=true;  }  else flag=false;  }while(flag);  enfermedades= Enfermedad();  lesiones= Lesion();  system("cls");  cpassword=Pass();  system("cls");  id=users.size();  User us(name, id,cpassword, altura, lesiones, peso, enfermedades, edad, sex);  users.push\_back(us);  MostrarDatos(us);  system("pause");  Cambios(id);  system("cls");  cout<<"Inicie sesion"<<endl;  system("pause");  saveUsersToFile(users, "usuariosdata.txt");} |

|  |
| --- |
| //Funcion para que realice su loggin el usuario.  // 31 lineas de codigo  void Login\_user()  {  string nombre;  string password;  bool band=true;  bool flag=true;  nombre=OnlyLetters("Nombre");  for(int i=0; i<users.size();i++)  {  if(nombre==users[i].getName() )  {  do  {  cout<<"Password"<<endl;  getline(cin, password);  if(password!=users[i].getPassword())  {  cout<<"Contrasenia incorrecta, vuleve a introducir tu clave"<<endl;  system("pause");  system("cls");  }  else  band=false;  }while(band);  Cambios(i);  break;  }  if(i==users.size()-1)  {  cout<<"Usuario no encontrado"<<endl;  system("pause");  }  }  } |

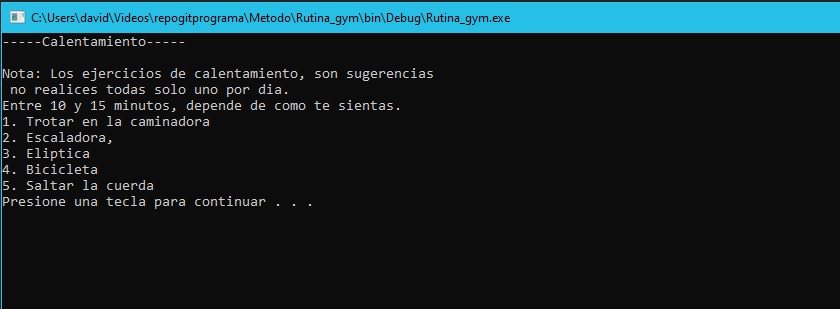
|  |
| --- |
| //Muestra el menu de inicio  //14 lineas de codigo  char Menu\_Inicio()  {  int lm=0;  string lmf;  char logs;  do  {  system("cls");  cout<<"--------Rutina de Gym--------\n\n\r"<<endl;  cout<<"1.- Usuario existente\n"<< "2.- nuevo usuario\n"<<"3.-salir\n"<<endl;  getline(cin, lmf);  }while(OpcionesValidas(lmf, 52));  lm=std::stoi(lmf);  return lm;  } |

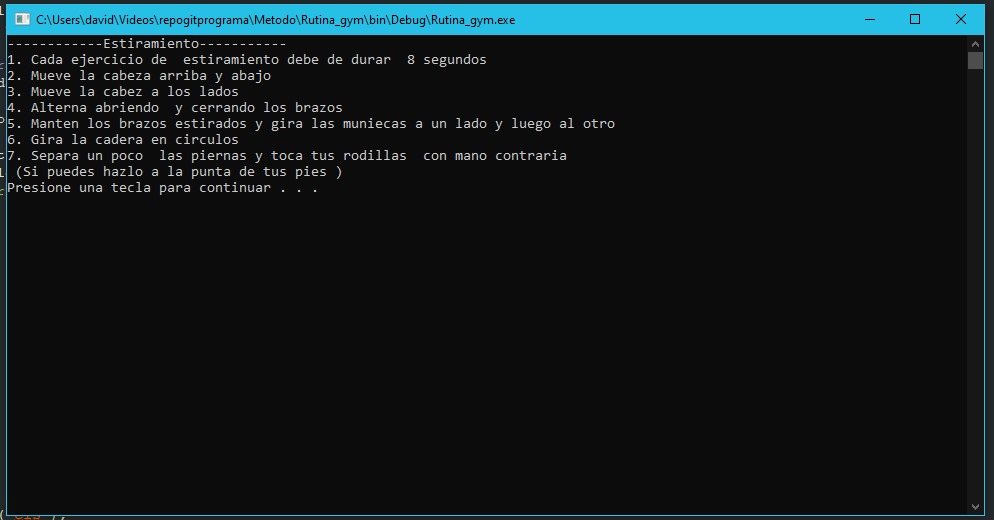
|  |
| --- |
| //Funcion main del codigo, empieza a leer un archivo. da 3 opciones para el usuario, y una cuarta para hacer testeos del codgio, solo para el programador.  int main()  {  bool reps=true;  readUsersFromFile("usuariosdata.txt");  do{  switch (Menu\_Inicio())  {  case 1:  {  system("cls");  Login\_user();  }  break;  case 2:  {  system("cls");  Registro\_User();  }  break;  case 3:  {  reps=false;  }  break;  case 4:  {  Gym\_day(0);  /\* int a = rand() % 10;  cout<<a;  system("pause");\*/  //MostrarDatos(users[0]);  system("pause");  }  break;  }  }  while(reps);  return 0;  } |

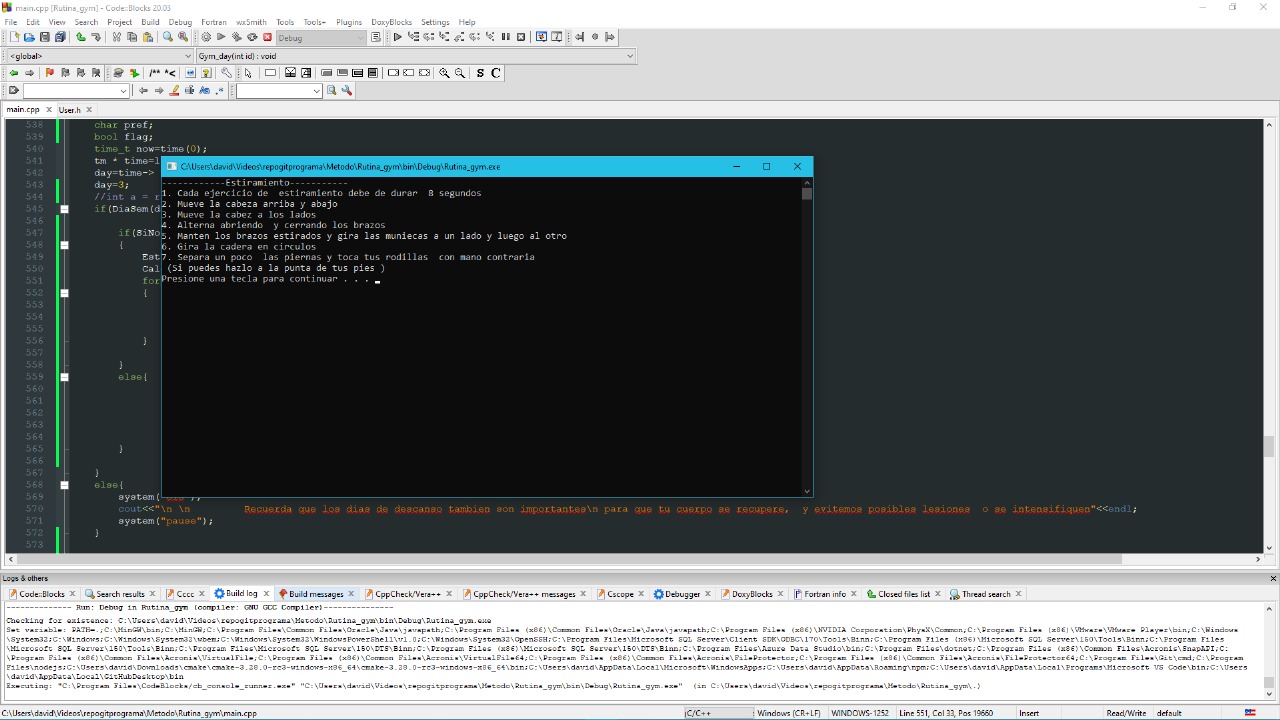
De preferencia, añadir número de líneas; en caso de ser muy extenso el código, tratar de describirlo por secciones para permitir su legibilidad

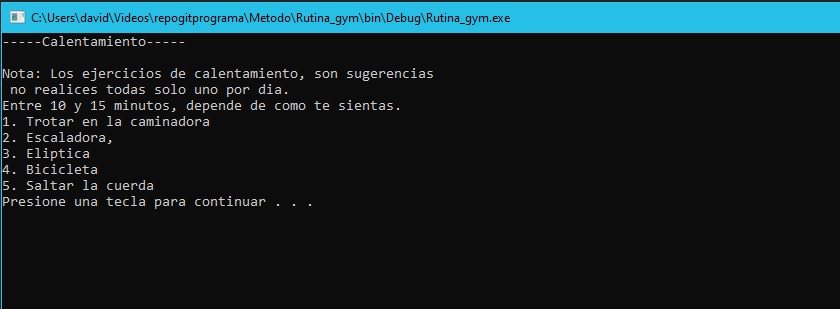
# Pruebas

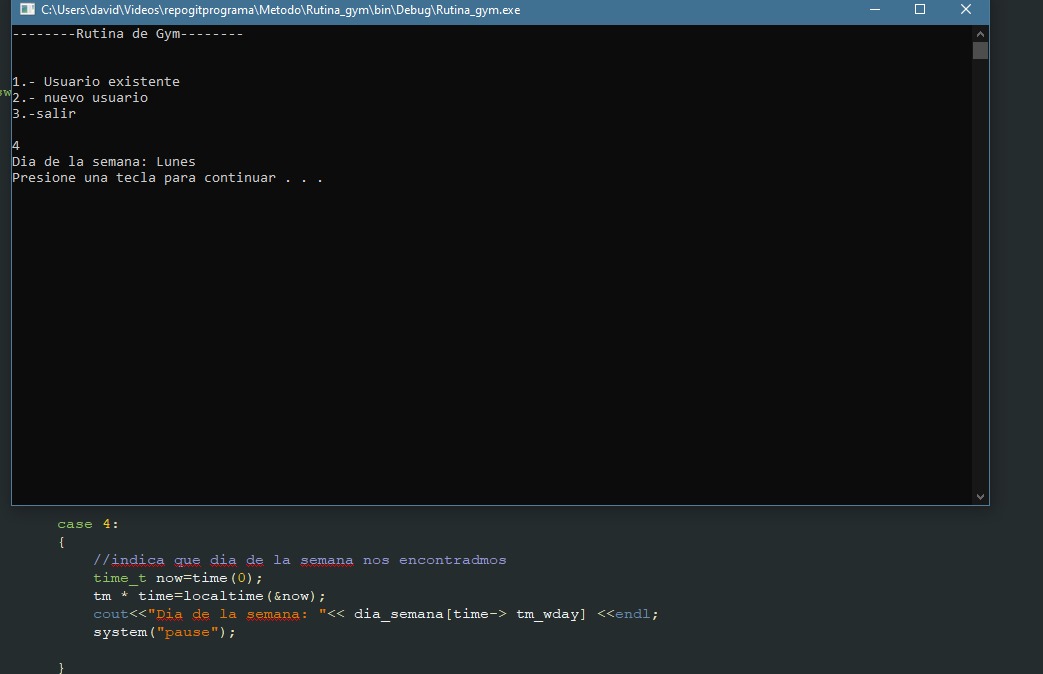
En esta sección deberán añadir las pruebas que han realizado a su proyecto: esto incluye introducir datos, capturas de pantalla de los resultados.





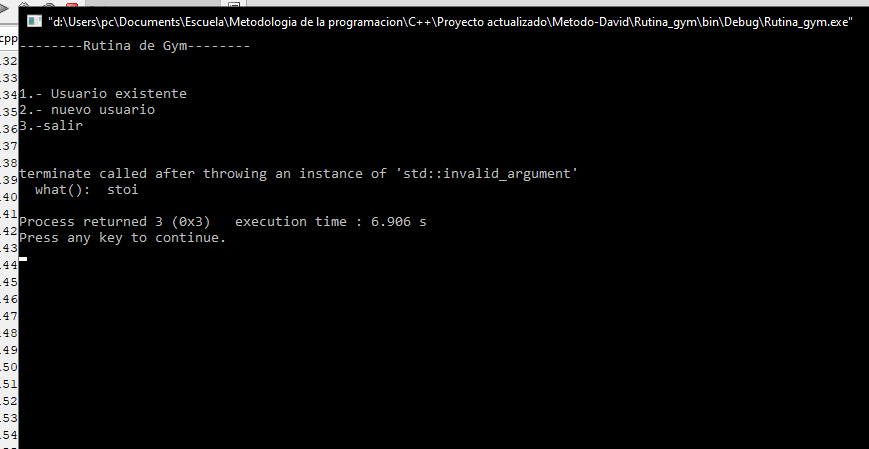






# Testeo para saber qué día de la semana nos encontramos

También se puede incluir capturas de pantalla de la consola mostrando pruebas fallidas.



# Falla: Al presionar “enter” el programa deja de funcionar, por lo cual se debe reiniciar.

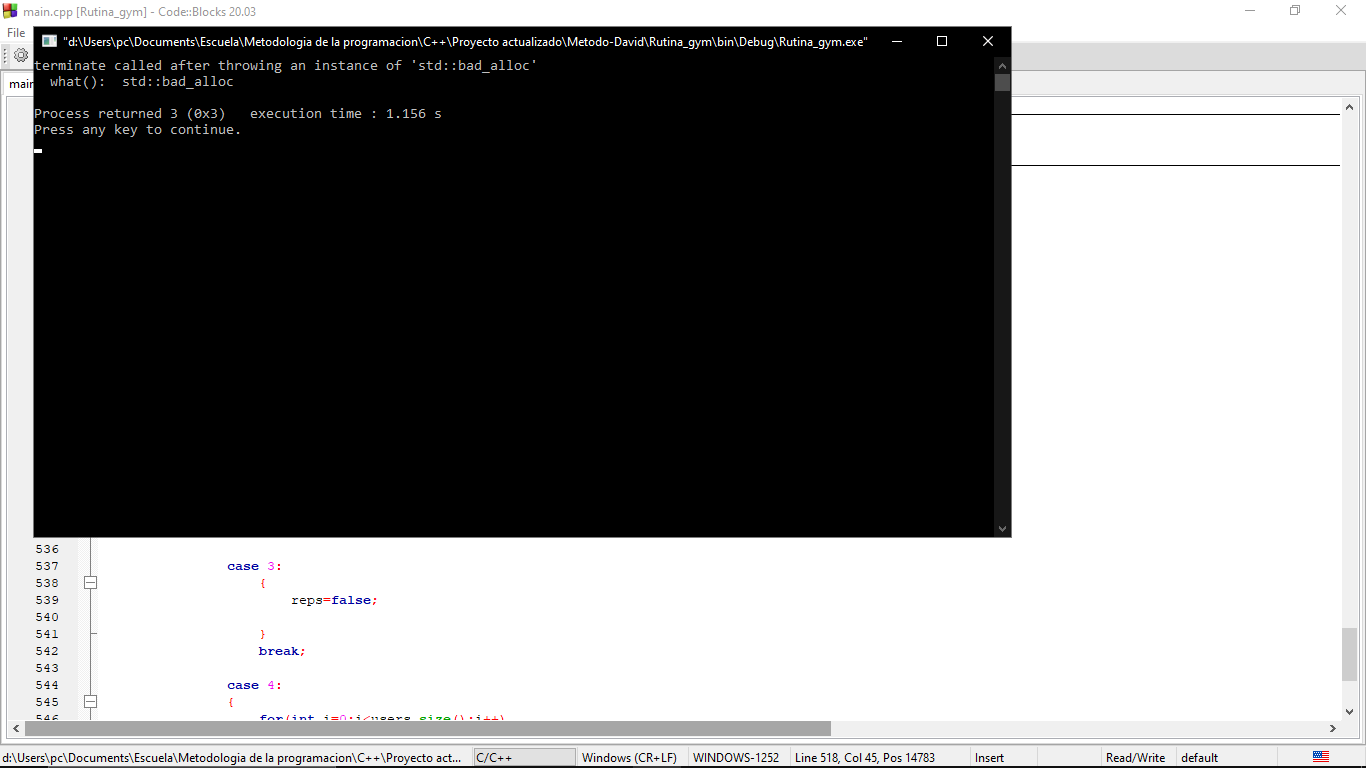
Solución:

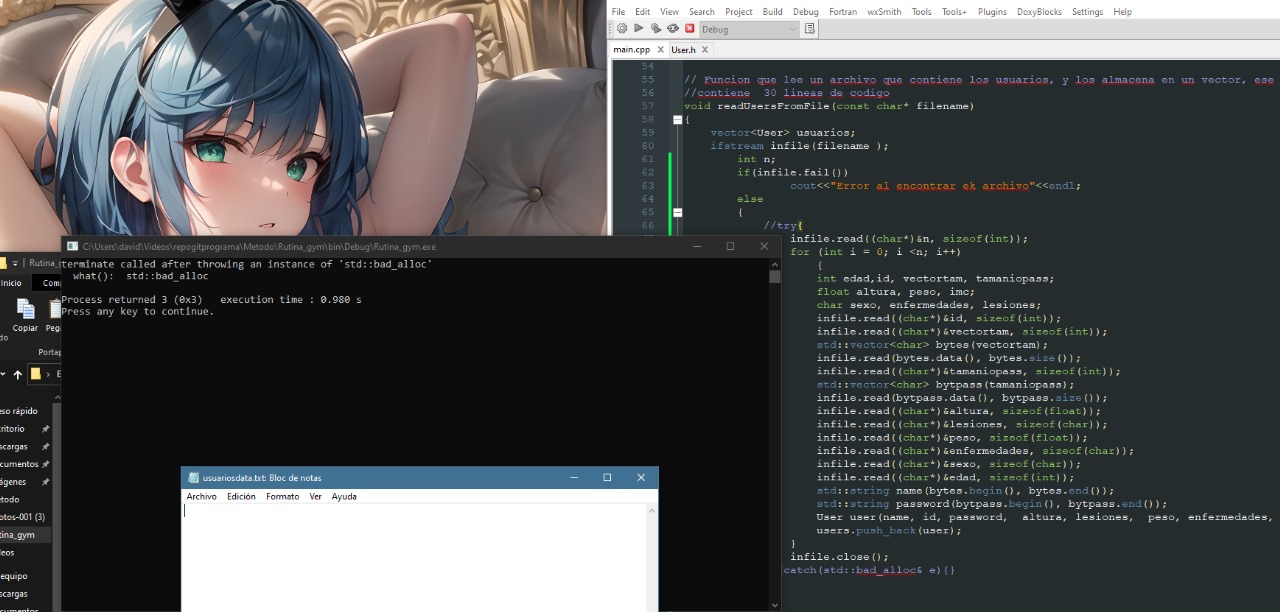
“try{

lm = int(logs);

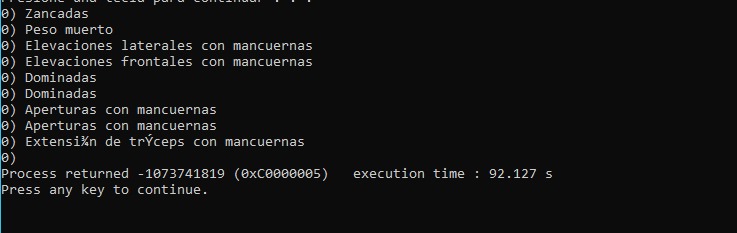
}catch(std::exception& e){cout<<"Nose guardo correctamente"<<endl;}”

Error cuando no encontraba el archivo de usuarios





error cuando no encuentra nada el archivo.



# Solucion

# for(int i=0; i<6;i++)

# {

# if(i!=5)

# cout<<(i+1)<<") "<<pesolibre[a + rand() % 5]<<endl;

# else cout<<(i+1)<<") "<<pesolibre[rand() % 26]<<endl;

# a=a+6;

# }

# Resultados finales

Captura de pantalla de computadora

Descripción generada automáticamente Para poder presentar resultado finales registramos algunos usuarios para que se guarden en un archivo de texto y al ejecutar el programa este los lea de ese archivo.

Texto

Descripción generada automáticamentePasaremos al menú de inicio del cual es un loggin para que el usuario se pueda registrar o si esta registrado solo entre a su cuenta.

Texto

Descripción generada automáticamenteSeguimos con la opcion 1 que es que ya esta registrado el usuario del cual nos pedira en primer instancia nuestro nombre de usuario y la contraseña del usuario.

Después nos mostrara nuestra información que habíamos registrado previamente y nos preguntara si deseamos realizar algún cambio de nuestra información.

Texto

Descripción generada automáticamente

En caso de que si la quiera cambiar, le desplegara un menú de opciones a cambiar su información, esto estará en un ciclo hasta que el usuario decida ya no cambiar nada

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Después nos preguntara si queremos trabajar con peso libre.

Texto

Descripción generada automáticamente

Si el usuario inidica que si, nos pondrá ejercicios que solo contengan peso libre, pero antes de eso viene una lista de estiramientos y un pequeño calentamiento.

Texto

Descripción generada automáticamenteTexto

Descripción generada automáticamenteY por ultimo nos dará la rutina para el gym, dependiendo si tenemos alguna enfermedad o o lesión.

(rutina si decidimos trabajar con peso libre).

Texto

Descripción generada automáticamente

(rutina que nos aportara desde maquinas del gym y ejercicios de peso libre)

Texto

Descripción generada automáticamente

Y al finalizar se cierra el programa.

Ahora ver cuando registramos usuarios.

# Conclusiones

Cada integrante deberá añadir su conclusión sobre el proyecto. Ejemplo:

Israel Trejo Hernández: Con este proyecto aprendí un poco de todo lo que puede hacer la programación y como aplicarla, con esta muestra aprendí que puede ser útil, que es herramienta muy versátil, que puede ayudar a gente y facilitar procesos en nuestro día a día o en procesos de empresas y grandes corporaciones de casi cualquier tipo. Además de que con el suficiente conocimiento se pueden hacer muchas cosas, incluso bromas y chistes.

Joshua Anwa Rodríguez Castellano: En conclusión, el algoritmo diseñado para la creación de rutinas de ejercicio ha demostrado ser una herramienta efectiva y versátil en la planificación personalizada de entrenamientos físicos. Su capacidad para adaptarse a las preferencias, objetivos y niveles de condición física de los usuarios lo convierte en una solución integral. La inclusión de variables como la frecuencia, intensidad y tipo de ejercicio garantiza una programación equilibrada y progresiva. Además, la interfaz intuitiva facilita su uso, permitiendo a los usuarios acceder de manera fácil a rutinas bien estructuradas. En resumen, este algoritmo no solo optimiza la eficiencia del entrenamiento, sino que también promueve la adherencia a un estilo de vida activo al brindar experiencias personalizadas y motivadoras.

José Emiliano Cabrera Villarreal: En este trabajo aprendí sobre funciones más que otras cosas, y como aplicarla, la utilidad que tiene a la hora de reducir memoria, facilita los procesos para obtener un valor dentro de esta misma, hasta ahora con este trabajo comprendí un poco de lo que es la programación. Por otro lado, durante la búsqueda de información sobre nuestro tema, encontré varios ejercicios que no conocía, y de alguna manera también dialogamos de cómo se debería administrar estos ejercicios para la gente novata.

Ricardo Xochitla Guzmán: En este trabajo se ha mostrado que nosotros como futuros programadores podemos crear software que beneficie a muchas personas. Éste, por ejemplo, ha sido un pequeño proyecto en el que tratamos de ayudar a las personas que tienen algún problema de salud a intentar mejorar y ponerse en forma. Lo hemos realizado siendo principiantes en el tema, por lo que todo lo que podríamos crear con algo más de experiencia y conocimiento es emocionante, así que debemos tener la capacidad de usar todo lo que aprendemos durante esta carrera para tratar de ayudar a las personas. Por último, cabe mencionar que hacer un proyecto de programación por más pequeño que sea requiere de mucho trabajo y frustración por lo que la paciencia es la clave.

David Carmona de Jesus: En este trabajo pudimos darnos cuenta, de lo que podemos realizar como programadores, desde un código muy sencillo a llegar a un código mas complejo, como también las posibles respuestas que pueden aportar los usuarios del programa, no obstante tenemos que ver mas allá, por cualquier error que se pueda generar o algo que no tengamos contemplado. Pudimos hacer uso de varios métodos, funciones, de manipulación de archivos .

# Referencias

# IEEE, “Men’s Health”, [Online]. Available: <https://www.menshealth.com/es/fitness/a33426021/entrenamiento-mancuernas-rutinas/>

IEEE, “Mayo Clinic”, [Online]. Available:<https://www.mayoclinic.org/healthy-lifestyle/fitness/basics/fitness-basics/hlv-20049447>

IEEE, “Mayo Clinic”, [Online]. Available: [www.mayoclinic.org/healthy-lifestyle/fitness/in-depth/walking-on-a-line/art-20047971](http://www.mayoclinic.org/healthy-lifestyle/fitness/in-depth/walking-on-a-line/art-20047971).

IEEE, “Mayo Clinic”, [Online]. Available: <https://www.mayoclinic.org/healthy-lifestyle/fitness/basics/aerobic-exercise/hlv-20049447>

IEEE, “Mayo Clinic”, [Online]. Available: <https://www.mayoclinic.org/healthy-lifestyle/fitness/basics/strength-training/hlv-20049447>

IEEE, “Mayo Clinic”, [Online]. Available: [www.mayoclinic.org/healthy-lifestyle/fitness/in-depth/squats/art-20046542](http://www.mayoclinic.org/healthy-lifestyle/fitness/in-depth/squats/art-20046542).

IEEE, “Mayo Clinic”, [Online]. Available: [www.mayoclinic.org/healthy-lifestyle/fitness/in-depth/leg-lifts/art-20047968](http://www.mayoclinic.org/healthy-lifestyle/fitness/in-depth/leg-lifts/art-20047968).

IEEE, “Mayo Clinic”, [Online]. Available: [www.mayoclinic.org/healthy-lifestyle/fitness/in-depth/push-ups/art-20047969](http://www.mayoclinic.org/healthy-lifestyle/fitness/in-depth/push-ups/art-20047969).

IEEE, “Mayo Clinic”, [Online]. Available: [www.mayoclinic.org/healthy-lifestyle/fitness/in-depth/single-leg-balance/art-20047970](http://www.mayoclinic.org/healthy-lifestyle/fitness/in-depth/single-leg-balance/art-20047970).

IEEE, “Mayo Clinic”, [Online]. Available: [www.mayoclinic.org/healthy-lifestyle/adult-health/in-depth/breathing-exercises/art-20045780](http://www.mayoclinic.org/healthy-lifestyle/adult-health/in-depth/breathing-exercises/art-20045780).

IEEE, “Mayo Clinic”, [Online]. Available: [www.mayoclinic.org/healthy-lifestyle/adult-health/in-depth/breathing-exercises/art-20045781](http://www.mayoclinic.org/healthy-lifestyle/adult-health/in-depth/breathing-exercises/art-20045781).

IEEE, “Mayo Clinic”, [Online]. Available: [www.mayoclinic.org/healthy-lifestyle/adult-health/in-depth/](http://www.mayoclinic.org/healthy-lifestyle/adult-health/in-depth/)

Políticas de evaluación

En este apartado se mostrarán las políticas de evaluación del proyecto final. Se muestran los conceptos que deben cubrir cada integrante para tener derecho a una calificación favorable.

# Evaluación

El proyecto se evaluará de la siguiente forma:

## Calidad del reporte:

El reporte no deberá contener faltas de ortografía, las imágenes deben ser de buena calidad, la redacción debe ser aceptable: no añadir información innecesaria.

## Evaluación individual del código

Cada integrante será evaluado sobre una parte al azar del código y deberá poder describirla congruentemente.

## Funcionamiento del proyecto

Se evaluará si el proyecto funciona adecuadamente.

## Presentación del proyecto

Cada equipo tendrá 15 minutos para presentar su proyecto. Se recomienda la siguiente organización del tiempo:

* 5 minutos para describir brevemente el proyecto
* 5 minutos para mostrar pruebas
* 5 minutos de evaluación individual y entrega de proyecto

# Evaluación de plagio

Se utilizarán diversas herramientas anti-plagio para detectar cualquier tipo de violación de derechos de autor en la escritura del código y del reporte:

* <https://unicheck.com/>
* <https://www.grammarly.com/plagiarism-checker>
* <https://gptzero.me/>
* Internet

En caso de superar el 30% de plagio se aplicará una penalización de 35% sobre la calificación final: si la calificación del proyecto es de 100% pero se detectó plagio, entonces será de 65%.

En caso de superar el 40% de plagio se aplicará una penalización de 60% sobre la calificación final: si la calificación del proyecto es de 100% pero se detectó plagio, entonces será de 40%.

En caso de superar el 50% de plagio se aplicará una penalización de 90% sobre la calificación final: si la calificación del proyecto es de 100% pero se detectó plagio, entonces será de 10%.

En caso de superar el 60% o más de plagio se aplicará una penalización de 100% sobre la calificación final: si la calificación del proyecto es de 100% pero se detectó plagio, entonces será de 0%.

# Entrega de archivos del proyecto

Los archivos del proyecto de deberán enviar al siguiente correo:

[carlos.carreon@correo.buap.mx](mailto:carlos.carreon@correo.buap.mx)

Se deberá comprimir los archivos del proyecto y se deberá añadir la siguiente contraseña:

1234

Recuerda, añadir cualquier archivo malicioso al proyecto significa dañar el patrimonio universitario (el dominio @correo.buap.mx es un patrimonio de la BUAP)

**ARTÍCULO 8**.- Para efectos de este ordenamiento incurre en responsabilidad patrimonial la autoridad o funcionario universitario que incumpla con alguna de sus obligaciones señaladas en el artículo 6, o realice actos u omisiones que redunden en daño o perjuicio al patrimonio universitario, lo que originará la instauración del procedimiento administrativo de responsabilidad patrimonial y en su caso, la aplicación de las sanciones correspondientes.

**ARTÍCULO 9**.- Las responsabilidades se fincarán a los universitarios que directa o indirectamente hayan ejecutado los actos o incurrido en omisiones que afecten el patrimonio universitario.

**ARTÍCULO 33**.- El monto de las sanciones económicas impuestas se establecerá considerando los daños y perjuicios causados al patrimonio universitario.

Artículos extraídos del siguiente documento: ACUERDO POR EL QUE SE ESTABLECEN LINEAMIENTOS SOBRE RESPONSABILIDAD PATRIMONIAL DE LA BENEMÉRITA UNIVERSIDAD AUTÓNOMA DE PUEBLA.