

Postmortem Baja Planes.

10/05/17

0.1

|  |  |
| --- | --- |
| **Nombre.** | **Matricula** |
| **Jesús Alberto Goiz Barrales.** | **24400085** |
| **Juan Alberto Gutiérrez canto.** | **24400063** |
| **Brenda Robles Antonio.** | **24400073** |
| **Rene Moratilla Montes.** | **24400075** |
| **Guillermo Vivaldo Vazquez** | **24400093** |

**Control De Versiones**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nombre Del Archivo | Versión | Fecha | Autor | Comentarios |
| Csoft-PM\_Baja\_Planes-100517.docx | 0.1 | 10/05/17 | JAGB | Creación del documento. |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

# Índice

[**Índice**](#_kd5bfhkalq0n) **2**

[**I.- Forma PSP Project Plan Summary.**](#_jhy44tc5k06j) **3**

[**II.- Time Recording Log.**](#_7e50s7cf9wc3) **9**

[**III.- Defect Recording Log.**](#_3znysh7) **10**

[**IV.- Diseño.**](#_m4hw86ggwsmr) **13**

[**V.- Design CheckList.**](#_92tu2pewgguj) **18**

[**VI.- Programa Fuente.**](#_6bvnqqrwimi9) **20**

[**VII.- Code CheckList.**](#_87ns93g4124c) **24**

[**VIII.- Reporte de resultados.**](#_bo7s7l2hg3ra) **27**

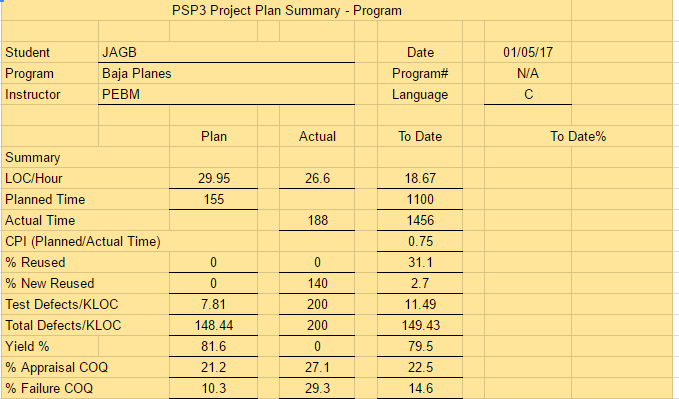
[**Template de pruebas**](#_1fob9te) **27**

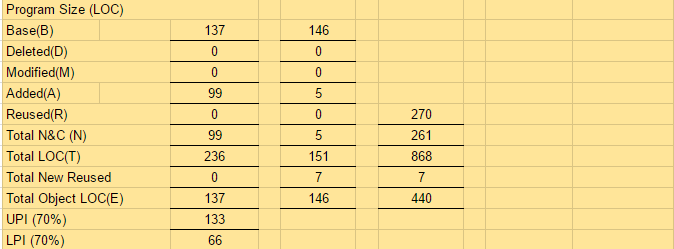
[**IX.- PIP**](#_4ocjjwajumc0) **28**

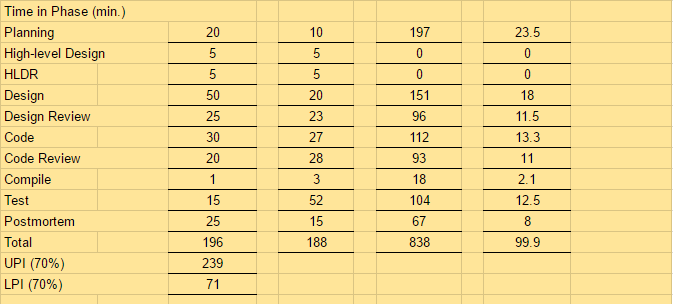
[**X.- Size Estimating Template**](#_w1ych8jaj7as) **29**

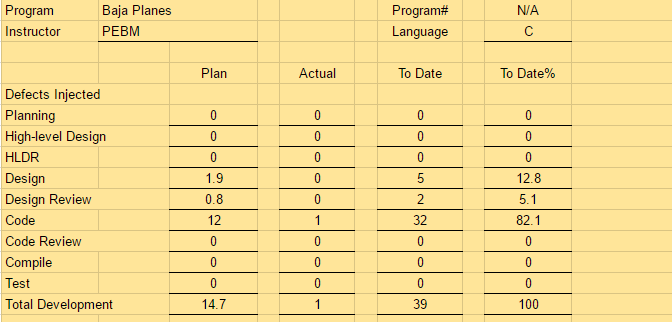
[**XI.- Task and Schedule**](#_195jonwtvlxs) **32**

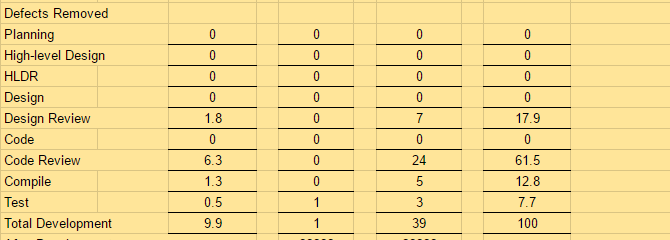
# I.- Forma PSP Project Plan Summary.

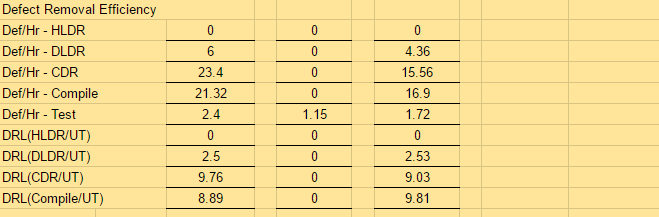




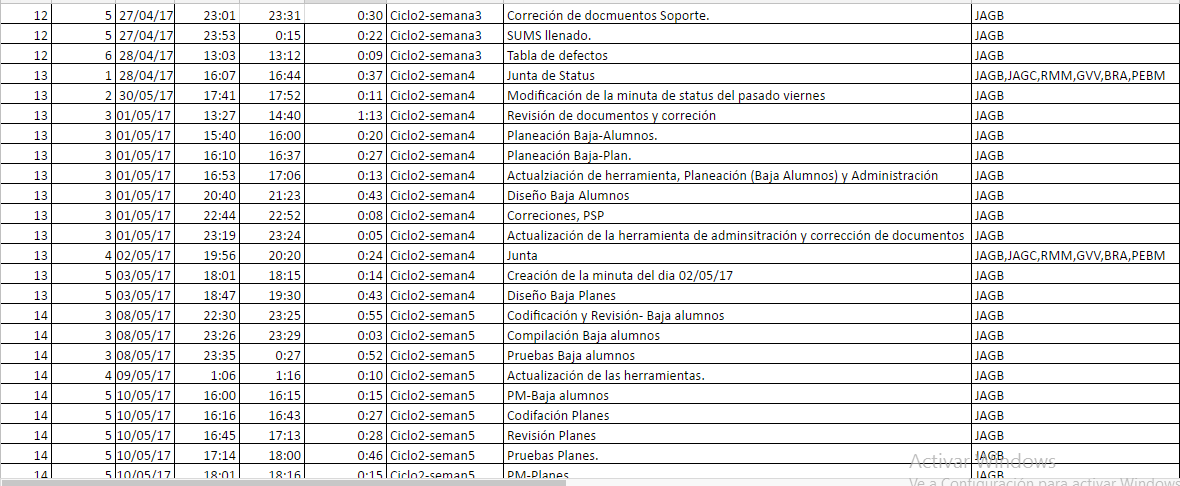




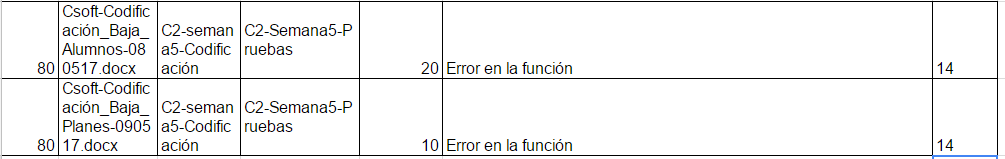




# II.- Time Recording Log.



# III.- Defect Recording Log.



# IV.- Diseño.

# State Specification Template

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Student | | JAGB | | | Date | | 03/05/17 |
| Program | | Baja Planes | | | Program # | | N/A |
| Instructor | | PEBM | | | Language | | C++ |
|  | | |  | | | | |
| Cambios planes | | | | | | | |
| **State Name** | | | | **Description** | | | |
| Inicio | | | | Inicia el programa. | | | |
| Baja planes | | | | Función principal para la baja de planes. | | | |
| Query planes | | | | Consulta a base de datos de planes | | | |
| planes | | | | Función para hacer borrar en la base de datos de planes.. | | | |
| Salir | | | | Termina el programa | | | |
|  | | | |  | | | |
| **Function/Parameter** | | | | **Description** | | | |
| printf() | | | | Imprime datos en la pantalla | | | |
| gets() | | | | Lee datos de la pantalla | | | |
| scanf() | | | | Lee datos de la pantalla | | | |
| opción | | | | opción que elegirá el usuario | | | |
| baja\_planes() | | | | Función principal de cambios | | | |
| query\_planes() | | | | Función de consulta | | | |
| delete\_planes() | | | | Función de borrado en base de datos | | | |
| **States/Next States** | | | | **Transition Condition** | | **Action** | |
| Inicio | | | |  | |  | |
|  | Baja\_planes | | | True | | printf()-imprime el menú  scanf()- lee opción  gets()- condición de datos válidos. | |
| Cambios planes | | | |  | |  | |
|  | Query\_planes | | | buscar un plan | | query\_planes() | |
|  | Delete\_planes | | | Borrar el registro de un plan. | | delete\_planes() | |
|  |  | | |  | |  | |

# Operational Specification Template

|  |  |  |  |
| --- | --- | --- | --- |
| **Student** | JAGB | **Date** | 03/05/17 |
| **Program** | Bajas Planes | **Program #** | N/A |
| **Instructor** | PEBM | **Language** | C++ |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Scenario Number** | **1** | **User Objective** | Dar de baja un plan. | |
| **Scenario Objective** | | Borrar un plan, opción Si | | |
| **Source** | **Step** | **Action** | | **Comments** |
| Usuario | 1 | Abrir el programa. | |  |
| Programa | 2 | Se muestra el menú de planen la opción de dar de baja. | | Menú de planes de alta, baja y modificar. |
| Usuario | 3 | Selecciona la opción de Baja. | | La opción 2 es para bajas. |
| Programa | 4 | Pide el id del plan | | Este id es el del plan que se desea cambiar los datos. |
| Usuario | 5 | Introduce el id del plan | | Para hacer el query del plan |
| Programa | 6 | Leer opción del usuario | | Verificar que sea valida |
| Programa | 7 | Muestra los datos del plan | | Imprime en pantalla los datos del plan |
| Programa | 8 | Se le pregunta al usuario si desea eliminar los datos que se muestran en pantalla del plan. | | Al seleccionar solo son dos opciones “Si” y “No” |
| Usuario | 8 | Introduce la opción de Si- | |  |
| Programa | 9 | Muestra un mensaje de “ Datos eliminados con éxito” y regrese al menú de planes. | |  |
| Usuario | 10 | Visualiza el mensaje de confirmación y regresa al menú principal de planes. | |  |
| Programa | 11 | Fin de la función Baja plan | |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Scenario Number** | **1** | **User Objective** | Dar de baja un plan. | |
| **Scenario Objective** | | Borrar un plan, opción No | | |
| **Source** | **Step** | **Action** | | **Comments** |
| Usuario | 1 | Abrir el programa. | |  |
| Programa | 2 | Se muestra el menú de planes en la opción de dar de baja. | | Menú de planes de alta, baja y modificar. |
| Usuario | 3 | Selecciona la opción de Baja. | | La opción 2 es para bajas. |
| Programa | 4 | Pide el id del plan | | Este id es el del plan que se desea cambiar los datos. |
| Usuario | 5 | Introduce el id del plan | | Para hacer el query del plan |
| Programa | 6 | Leer opción del usuario | | Verificar que sea valida |
| Programa | 7 | Muestra los datos del plan | | Imprime en pantalla los datos del plan |
| Programa | 8 | Se le pregunta al usuario si desea eliminar los datos que se muestran en pantalla del plan. | | Al seleccionar solo son dos opciones “Si” y “No” |
| Usuario | 8 | Introduce la opción de “No” | |  |
| Programa | 9 | Muestra un mensaje de alerta “ Los datos no serán eliminados” y regresa al menú principal del plan. | |  |
| Usuario | 10 | Visualiza el mensaje de alerta del programa y este mismo lo lleva al menú principal de los planes.. | |  |
| Programa | 11 | Fin de la función Baja plan | |  |

# Logic Specification Template

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Student | JAGB | | Date | 01/05/17 |
| Program | Baja Planes | | Program # | N/A |
| Instructor | PEBM | | Language | C++ |
| Object | N/A | | Function | baja\_planes() |
| **INCLUDES:**  **TYPE DEFINITIONS:**  **Declaration:** | | #include <cstdio>  #include <iostream>  #include "sqlite3.h"  Using namespace std; | | |
|  | |  | | |

|  |  |
| --- | --- |
| **Reference:** |  |

|  |  |
| --- | --- |
| Logic reference numbers | Program logic, in pseudocode |
| 1 | Imprime la opción de insertar ID del plan para realizar la búsqueda. |
| 2 | Se manda a llamar la función de búsqueda para imprimir los datos en pantalla del plan. |
| 3 | Se presenta la opción de borrar plan con un “si” o “no”  Si es si: se manda a llamar la función de delete para borrar el registro del planes.  Si es no: Se regresa al menú principal de los planes(alta, borrar, modificar). |
|  | Fin. |

# 

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Object | N/A | | Function | query\_planes() |
| **INCLUDES:**  **TYPE DEFINITIONS:**  **Declaration:** | | #include <cstdio>  #include <iostream>  #include "sqlite3.h"  Using namespace std; | | |
|  | |  | | |

|  |  |
| --- | --- |
| **Reference:** |  |

|  |  |
| --- | --- |
| Logic reference numbers | Program logic, in pseudocode |
| 1 | Asignados a sql=”SELECT \* FROM PLANES where Id\_Al=@Id\_A; |
| 2 | Asignamos a rc el resultado de preparar la sentencia sql |
| 3 | En caso de que este mal |
| 4 | imprime el error |
| 5 | En caso de que sea correcta |
| 6 | Se busca el índice del identificador a cambiar |
| 7 | Se cambia el valor por el original |
| 8 | Se comprueba cuántos resultados existen |
| 9 | En caso de que exista uno |
| 10 | Se imprimen los valores |
| 11 | Se destruye el objeto |
| 12 | Fin. |

# 

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Object | N/A | | Function | delete\_planes() |
| **INCLUDES:**  **TYPE DEFINITIONS:**  **Declaration:** | | #include <cstdio>  #include <iostream>  #include "sqlite3.h"  Using namespace std; | | |
|  | |  | | |

|  |  |
| --- | --- |
| **Reference:** |  |

|  |  |
| --- | --- |
| Logic reference numbers | Program logic, in pseudocode |
| 1 | Asignados a sql=”Delete PLANES SET @Type=@Value where Id\_Al=@Id\_A; |
| 2 | Asignamos a rc el resultado de preparar la sentencia sql |
| 3 | En caso de que este mal |
| 4 | imprime el error |
| 5 | En caso de que sea correcta |
| 6 | Se busca el índice de los identificadores a cambiar |
| 7 | Se cambia el valor por el original |
| 8 | Se comprueba cuántos resultados existen |
| 9 | En caso de que exista uno |
| 10 | Se imprimen los valores |
| 11 | Se destruye el objeto |
| 12 | Fin. |

# Functional Specification Template

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Student** | | | JAGB | | **Date** | 03/05/17 |
| **Program** | | | Baja Planes | | **Program #** | N/A |
| **Instructor** | | | PEBM | | **Language** | C++ |
|  | | | | | | |
| **Class Name** | | void baja\_planes() | | | | |
| **Parent Class** | | N/A | | | | |
|  | | | | | | |
| **Attributes** | | | | | | |
|  | **Declaration** | | | **Description** | | |
|  | char opcion | | | Corresponde a la opción que el usuario eligió. | | |
|  | char valor [40] | | | Variable para el valor del nombre. | | |
|  | sqlite3 \*db | | | Base de datos | | |
|  | | | | | | |
| **Items** | | | | | | |
|  | **Declaration** | | | **Description** | | |
|  | query\_planes() | | | Consulta de la base de datos | | |
|  | delete\_planess() | | | Cambio directo a la base de datos | | |
|  | printf() | | | Para imprimir directamente en la interfaz de usuario. | | |
|  | cin() | | | Para leer datos de la interfaz de usuario. | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Class Name** | | void query\_planes() | | | | |
| **Parent Class** | | N/A | | | | |
|  | | | | | | |
| **Attributes** | | | | | | |
|  | **Declaration** | | | **Description** | | |
|  | sqlite3 \*db | | | Base de datos | | |
|  | char\* id | | | Id del plan | | |
|  | char \*sql | | | Para la sentencia sql | | |
|  | char \*err | | | Para el error de la consulta sql | | |
|  | int rc | | | Valor de consulta sql | | |
|  | int step | | | Número de resultados | | |
|  | sqlite3\_stmt \*res | | | Resultado de consulta sql | | |
|  | | | | | | |
| **Items** | | | | | | |
|  | **Declaration** | | | **Description** | | |
|  | sqlite3\_prepare\_v2() | | | Prepara la sentencia sql para la consulta | | |
|  | SQLITE\_OK | | | Para saber si es correcta la sentencia sql | | |
|  | sqlite3\_bind\_parameter\_index() | | | Busca dónde está el identificador del parámetro a cambiar. | | |
|  | sqlite3\_bind\_text() | | | Inyecta un valor en la sentencia sql | | |
|  | sqlite3\_step() | | | Resultados de la consulta | | |
|  | SQLITE\_ROW | | | Para saber si hay datos | | |
|  | sqlite3\_column\_text() | | | Localiza el valor de la consulta en memoria | | |
|  | sqlite3\_finalize() | | | Finaliza el objeto creado. | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Class Name** | | void delete\_planes() | | | | |
| **Parent Class** | | N/A | | | | |
|  | | | | | | |
| **Attributes** | | | | | | |
|  | **Declaration** | | | **Description** | | |
|  | sqlite3 \*db | | | Base de datos | | |
|  | char\* id | | | Id del plan | | |
|  | char \*sql | | | Para la sentencia sql | | |
|  | char \*err | | | Para el error de la consulta sql | | |
|  | int rc | | | Valor de consulta sql | | |
|  | int step | | | Número de resultados | | |
|  | sqlite3\_stmt \*res | | | Resultado de consulta sql | | |
|  | char\* valor | | | Valor a cambiar | | |
|  | char\* type | | | Tipo de valor a cambiar | | |
|  | | | | | | |
| **Items** | | | | | | |
|  | **Declaration** | | | **Description** | | |
|  | sqlite3\_prepare\_v2() | | | Prepara la sentencia sql para la consulta | | |
|  | SQLITE\_OK | | | Para saber si es correcta la sentencia sql | | |
|  | sqlite3\_bind\_parameter\_index() | | | Busca dónde está el identificador del parámetro a cambiar. | | |
|  | sqlite3\_bind\_text() | | | Inyecta un valor en la sentencia sql | | |
|  | sqlite3\_step() | | | Resultados de la consulta | | |
|  | SQLITE\_ROW | | | Para saber si hay datos | | |
|  | sqlite3\_column\_text() | | | Localiza el valor de la consulta en memoria | | |
|  | sqlite3\_finalize() | | | Finaliza el objeto creado. | | |

# V.- Design CheckList.

**PSP2 Design Review Checklist**

|  |  |  |  |
| --- | --- | --- | --- |
| Student | JAGB | Date | 01/05/17 |
| Program | Baja Planes | Program # | N/A |
| Instructor | PEBM | Language | C++ |

|  |  |
| --- | --- |
| Purpose | To guide you in conducting an effective design review |
| General | - Review the entire program for each checklist category; do not attempt to review for more than one category at a time!  - As you complete each review step, check off that item in the box at the right.  - Complete the checklist for one program or program unit before reviewing the next. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Complete | Verify that the design covers all of the applicable requirements.  - All specified outputs are produced.  - All needed inputs are furnished.  - All required includes are stated. | MB  X | B    X  X | R | M |
| External Limits | Where the design assumes or relies upon external limits, determine if behavior is correct at nominal values, at limits, and beyond limits. |  |  |  |  |
| Logic | - Verify that program sequencing is proper.  Stacks, lists, and so on are in the proper order.  Recursion unwinds properly.  - Verify that all loops are properly initiated, incremented, and terminated.  - Examine each conditional statement and verify all cases. | X        X | X |  |  |
| Internal Limits | Where the design assumes or relies upon internal limits, determine if behavior is correct at nominal values, at limits, and beyond limits. |  |  |  |  |
| Special Cases | - Check all special cases.  - Ensure proper operation with empty, full, minimum, maximum, negative, and zero values for all variables.  - Protect against out-of-limits, overflow, and underflow conditions.  - Ensure “impossible” conditions are absolutely impossible.  - Handle all possible incorrect or error conditions. | X  X    X  X  X |  |  |  |
| Functional Use | - Verify that all functions, procedures, or methods are fully understood and properly used.  - Verify that all externally referenced abstractions are precisely defined. | X    X |  |  |  |
| System Considerations | - Verify that the program does not cause system limits to be exceeded.  - Verify that all security-sensitive data are from trusted sources.  - Verify that all safety conditions conform to the safety specifications. | X    X    X |  |  |  |
| Names | Verify that  - all special names are clear, defined, and authenticated  - the scopes of all variables and parameters are self-evident or defined  - all named items are used within their declared scopes | X  X  X |  |  |  |
| Standards | Ensure that the design conforms to all applicable design standards. | X |  |  |  |

# VI.- Programa Fuente.

/\*=========================================================\*/

/\*Autor:Jesús Alberto Goiz Barrales \*/

/\*Componente: Baja Planes \*/

/\*Fecha: 09/05/17 \*/

/\*Descripción: Codificación que muestra el borrado de los planes \*/

/\*Versión: 0.1 \*/

/\*=========================================================\*/

/\*==========================================================\*/

/\* El programa contiene lo siguiente: \*/

/\* void query\_planes(sqlite3 \*db,char\* id\_plan) \*/

/\* void borrar\_planes(sqlite3 \*db,char\* type,char\* value,char\* id\_plan) \*/

/\* void borrar\_posee(sqlite3 \*db,char\* type,char\* value,char\* id\_plan) \*/

/\* \*/

/\* int main() \*/

/\*==========================================================\*/

#include <cstdio>

#include <cstdlib>

#include <iostream>

#include "sqlite3.h"

#include "altas.h"

using namespace std;

/\*\*\*Esta función se encarga de borrar el plan en la base de datos\*\*\*/

void borrar\_planes(sqlite3 \*db,char\* id\_plan)

{

char \*sql; /\*Para la sentencia sql \*/

char \*err=0; /\*Para el mensaje de error de la sentencia sql \*/

int rc;

sqlite3\_stmt \*res;

sql="DELETE from Planes where ID LIKE @Id\_P;"; /\* Selecciona a los planes con un id especifico sentencia sql\*/

rc=sqlite3\_prepare\_v2(db,sql,-1,&res,0); /\* Verifica si es valida la instruccion sql que se solicita\*/

if(rc!=SQLITE\_OK){

printf("\t\t QUERY-SQL error %s\n",err); /\* Imprime el error de la base de datos\*/

}

else{

printf("\t\t Borrado!!!\n"); /\* Se aprueba la sentencia sql\*/

int idx = sqlite3\_bind\_parameter\_index(res, "@Id\_P"); /\* Se busca en que lugar esta el identificador a cambiar id\*/

sqlite3\_bind\_text(res, idx, id\_plan,40,SQLITE\_STATIC); /\* Se cambia el identificador por un valor determinado id\*/

}

int step = sqlite3\_step(res);

if (step == SQLITE\_ROW) { /\* indica el resultado de la instrucción sql, valores del plan\*/

printf("\t\t Borrado!!!\n"); /\* Se aprueba la sentencia sql ya echa\*/

}

sqlite3\_finalize(res); /\* Finaliza el objeto creado\*/

}//20

void borrar\_posee(sqlite3 \*db,char\* id\_plan)

{

char \*sql; /\*Para la sentencia sql \*/

char \*err=0; /\*Para el mensaje de error de la sentencia sql \*/

int rc;

sqlite3\_stmt \*res;

sql="DELETE from Posee where ID\_Plan LIKE @Id\_P;";

rc=sqlite3\_prepare\_v2(db,sql,-1,&res,0); /\* Verifica si es valida la instruccion sql que se solicita\*/

if(rc!=SQLITE\_OK){

printf("\t\t QUERY-SQL error %s\n",err); /\* Imprime el error de la base de datos\*/

}

else{

printf("\t\t Borrado!!!\n"); /\* Se aprueba la sentencia sql\*/

int idx = sqlite3\_bind\_parameter\_index(res, "@Id\_P"); /\* Se busca en que lugar esta el identificador a cambiar id\*/

sqlite3\_bind\_text(res, idx, id\_plan,40,SQLITE\_STATIC); /\* Se cambia el identificador por un valor determinado id\*/

}

int step = sqlite3\_step(res);

if (step == SQLITE\_ROW) { /\* indica el resultado de la instrucción sql, valores del plan\*/

printf("\t\t Borrado!!!\n"); /\* Se aprueba la sentencia sql ya echa\*/

}

sqlite3\_finalize(res); /\* Finaliza el objeto creado\*/

}

/\*\*\*Esta función es la funcion principal de borrado planes\*\*\*/

void baja\_planes(sqlite3 \*db){

char opcion;

char id[40];

do{

system("cls");

printf("\t\to----------------------------------------------------------o\n"); /\* Menú de borrar alumno\*/

printf("\t\t| ================ Borrar Planes =================== |\n");

printf("\t\to----------------------------------------------------------o\n");

printf("\t\to----------------------------------------------------------o\n");

printf("\t\t| ¿Que desea hacer? |\n");

printf("\t\t| 1)Buscar Plane |\n");

printf("\t\t| 2)Regresar Men%c Planes |\n",163);

printf("\t\t| Ingresa la opci%cn que desea: |\n",162);

printf("\t\to----------------------------------------------------------o\n\t\t");

cin>>opcion;

switch(opcion){

case '1':{

do{

printf("\t\t Ingresa el ID.-"); /\* Busca a un alumno para hacer borrar\*/

scanf("%s",id);

system("cls");

printf("\t\to----------------------------------------------------------o\n");

query\_planes(db,id);

printf("\t\to----------------------------------------------------------o\n"); /\* Imprime datos del alumno\*/

printf("\t\t| 0)Salir |\n");

printf("\t\t| Desea borrar a este Plan (Y/N)): |\n");

printf("\t\to----------------------------------------------------------o\n\t\t");

cin>>opcion;

switch(opcion){

case 'Y':{

// borrar\_planes(db,id);

// borrar\_posee(db,id);

break;

}

case 'N':{

printf("\t\t Regresando a men%c\n",163);

break;

}

case '0':{

printf("\t\t Regresando a men%c\n",163);

break;

}

default:{

printf("\t\t Valor incorrecto!!! intenta otro \n");

break;

}

}

}while(opcion!='0');

break;

}

case '2':{

printf("\t\t Regresando a men%c\n",163);

break;

}

default:{

printf("\t\t Valor incorrecto!!! intenta otro \n");

break;

}

}

}while(opcion !='2');

}//61

int main()

{

sqlite3 \*db; /\* Base de datos en sqlite3\*/

int res;

res=sqlite3\_open("testpm3f.db", &db); //query\_planes(db,"ISW001"); /\* Se abre la db\*/

baja\_planes(db); /\*A cambios alumnos\*/

return 0;

}

# VII.- Code CheckList.

**Code Review Checklist**

|  |  |  |  |
| --- | --- | --- | --- |
| Student | JAGB | Date | 10/05/17 |
| Program | Baja Planes | Program # | Baja Planes |
| Instructor | PEBM | Language | C |

|  |  |
| --- | --- |
| Purpose | To guide you in conducting an effective code review |
| General | - Review the entire program for each checklist category; do not attempt to review for more than one category at a time!  - As you complete each review step, check off that item in the box at the right.  - Complete the checklist for one program or program unit before reviewing the next. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Complete | Verify that the code covers all of the design. | MB | B | R | M |
| Includes | Verify that the includes are complete. | X |  |  |  |
| Initialization | Check variable and parameter initialization.  - at program initiation  - at start of every loop  - at class/function/procedure entry | x  X  X |  |  |  |
| Calls | Check function call formats.  - pointers  - parameters  - use of ‘&’ | X  X  X |  |  |  |
| Names | Check name spelling and use.  - Is it consistent?  - Is it within the declared scope?  - Do all structures and classes use ‘.’ reference? | x  X |  |  |  |
| Strings | Check that all strings are  - identified by pointers  - terminated by NULL | X  X |  |  |  |
| Pointers | Check that all  - pointers are initialized NULL  - pointers are deleted only after new  - new pointers are always deleted after use | X  X  X |  |  |  |
| Output Format | Check the output format.  - Line stepping is proper.  - Spacing is proper. | X  X |  |  |  |
| () Pairs | Ensure that () are proper and matched. |  | x |  |  |
| Logic Operators | - Verify the proper use of ==, =, ||, and so on.  - Check every logic function for (). | X  X |  |  |  |
| Line-by-line check | Check every line of code for  - instruction syntax  - proper punctuation |  | X  X |  |  |
| Standards | Ensure that the code conforms to the coding standards. | x |  |  |  |
| File Open and Close | Verify that all files are  - properly declared  - opened  - closed | X  X  X |  |  |  |

# VIII.- Reporte de resultados.

# Template de pruebas

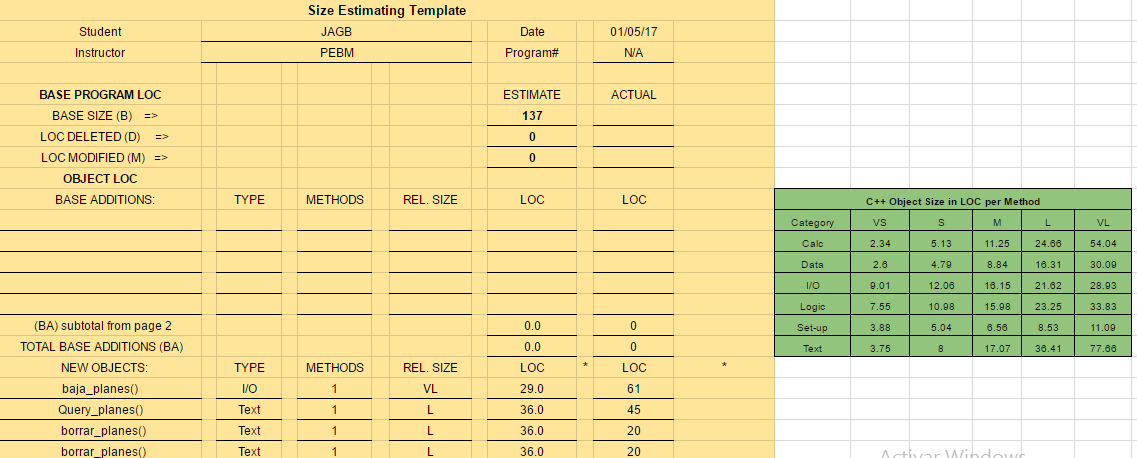
|  |  |
| --- | --- |
| Nombre de la Prueba | Prueba de baja de plan |
| Caso de Uso | 4.1 |
| Objetivo de la Prueba | Verificar la correcta eliminación de los datos de un maestro en la DB. |
| Descripción de la Prueba | Se ingresa como usuario Administrador y se realiza por medio de la opción de “Baja de Plan”. |
| Test | 1.-Direccionarse a la opción Baja Plan.  2.- Ingresar ID de Plan.  3.-Confirmación de eliminación. |
| Resultados Esperados | Mensaje de Éxito:  Datos Plan (Se hará consulta a DB)   1. Nombre de plan 2. Semestres 3. Materias   “Ha sido borrado ‘Plan´ de Sistema” (Si no se encuentra en Consulta a DB)  Mensaje de Error:  “No fue borrado de sistema” (Se encuentra en consulta a DB). |

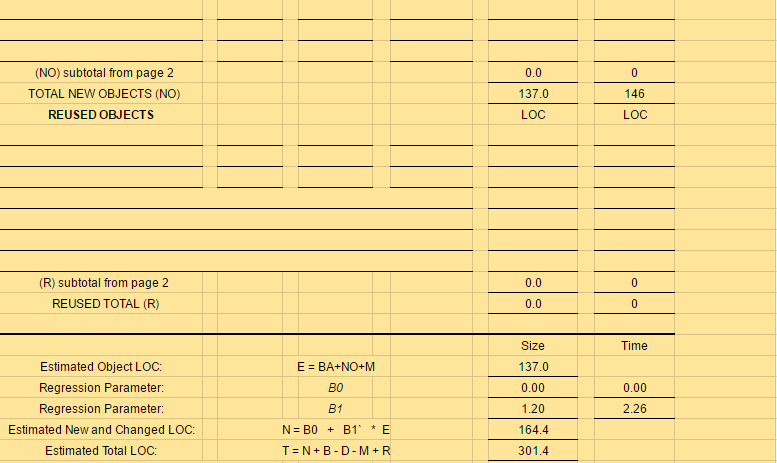
|  |  |
| --- | --- |
| Capturas De Pantalla | 1.PNG  2.PNG  3.PNG  4.PNG  5.PNG  6.PNG |
| Estado caso de prueba | Se realizó la baja correctamente. |
| Errores obtenidos | Los querys estaban mal diseñados y no imprimían los valores almacenados en la BD, los update se optimizaron porque no se actualizaban los campos indicados. |
| Responsable de prueba | JAGB |
| Comentarios | Sin comentarios |

# IX.- PIP

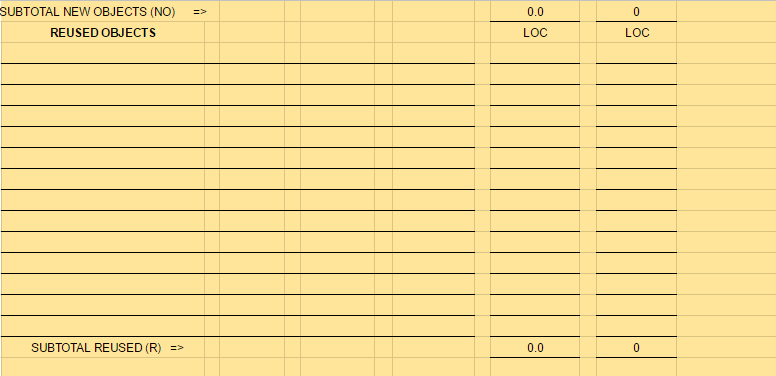
|  |  |  |
| --- | --- | --- |
| Menú-Alumnos | | |
| No. PIP | Descripción del Problema | Descripción de la propuesta. |
| 1 | Mala organición de ideas | Tener mejor panorama al momento de hacer la planeación de nuestro PSP |
| 2 | Confusión de nombres de módulos | Tener una mejor organización en cuestión de análisis del problema. |

# X.- Size Estimating Template









# XI.- Task and Schedule

