CIS605 Project 2

Name: Sample Rubric

* Note: Feedback may refer to a specific code line number ("LN"). To view line numbers in Visual Studio, navigate as follows: (Main Menu) > Tools > Options > (Options Box) > Text Editor > All Languages > check "Line Numbers"

To see the content processory (pages and the one processor of the content processor) The processor of the content proces					
General Supervisionality (pagelication of an all productions and an activation of the solutions and all the application of the project should compile year when the solutions and all the application of the project should compile a basis of the solutions and all the application of the project should compile a basis of the project should be project should compile a basis of the project should be project should	Topic	Expectations	Points Possible	Points Received	Specific Feedback
To Fine, Solitotion, Inches the proped procept any and position of the control and all the political has all the solitons and all the political has all th	General Funct	ionality (applicable to all problem sets)	r ossible	Received	
Solution, include the solution and all the applicable projects should compile for projects and solutions of the solution of th	Zip File,	, , , ,	Mandatory		
Foreign 6 has placed by enjoyed should compile compile on the process of the proc	Solution,	include the solution and all the applicable	to receive a		
clearly (no compile errors, but compile Reference and Extracting grave recomments could) Program should not crash white running Use chemical supportable stage groups by on form Use chemical stage stage groups by on form Use chemical stage stage groups by one form the groups	Projects	file(s)	grade		
Comprise A bug deshifty (no compile errors, but compile from promise a segmentary solid and creative training and promise and progress should not creative valve manners and progress are removed. All shall stops correct the contractive valve manners are prefixed comments and progress are removed. All shall stops correct to fix user internal progress are removed. All comments are prefixed comments are prefixed comments used to be business terms. Because of the progress are removed. All comments are prefixed comments and in clair business terms. Because of the progress are removed. All comments are prefixed comments are prefixed comments used to business terms. Because of the progress are removed. All comments are prefixed comments are prefixed comments are prefixed comments and progress are removed. All comments are prefixed comments and progress are named correctly and named in clear business terms. Methods and progress are named correctly and named in clear business terms. Methods and progress are named correctly and named in clear business terms. Methods and progress are named correctly and named in clear business terms. Methods and progress are named correctly and named in clear business terms. Methods and progress are named correctly and named in clear business terms. Methods and progress are named correctly and named in clear business terms. Methods and progress are named correctly and named in clear business terms. Methods and progress are named correctly and named in clear business terms. Methods and progress are named correctly and named in clear business terms. Methods and progress are named correctly and named in clear business terms. Methods and progress are named correctly and named in clear business terms. Metho	Proiects		-20% off of		
The control developed in solution of control work of the control developed in the control develo		4			
General Asstrateirs (applicable for oil problem sets) Ui elements align properly on form Ui elements align properly on form Ui elements align properly on form Ui elements agriculture of the control of control	free	warnings are sometimes okay)	total		
Use elements appropriets year/dependent personal processors and the content of the content of processors and the content of processors and the content of processors and the content of the content of processors and the content of th	General Aesth				
Use elements appropriately sized/faced Codd creativity while maintaining professionalism Look and red Friendly capitions and messages with proper spelling/grammar understandable ToolTight pas appropriate Message log entires scroil and display the most recent entry Justification as appropriate (e.g., numbers are right justified) All tab stops correct Keybard shortcuts as appropriate (e.g., numbers are right justified) All tab stops correct Keybard shortcuts as appropriate (e.g., numbers are right justified) Appropriate Accept/Cancel functionality. Corrected for an if the County of the Count	General Aesti				
Good creativity while maintaining professional im User firendly captions and messages with professional im User firendly captions and messages with professional importance of the control		9 1 1 1			
Use infriendly captions and messages with progress pelling/grammary Understandable FootTips as appropriate Message log entries critical and display the most recent entry Justification as appropriate (e.g. numbers are right justified) All tols blorgo correct Replaced Appropriate (e.g. numbers are right justified) Curror reset of fix use implate most progress of the customer lab fire and public override in the customer lab, here, when you ener't the customer lab input, it is not shown what you have enerted until you reach a soectific number of lette until the customer lab input, it is not shown what you have enerted until you reach a soectific number of lette until the customer lab input, it is not shown what you have enerted until you reach a soectific number of lette until the customer lab input, it is not shown what you have enerted until you reach a soectific number of lette until the customer lab input, it is not shown what you have enerted until you reach a soectific number of lette until the customer lab, here, when you enert the customer lab input, it is not shown what you have enerted until you reach a soectific number of lette until the customer lab, here, when you enert the customer lab, here, when you enert the customer lab input, it is not shown what you have enerted until you reach a soectific number of lette until the customer lab input, it is not shown what you have entered until you reach a soectific number of lette until the customer lab input, it is not shown what you have entered until you reach a soectific number of lette until the customer lab in the custom					0.5 should have enough spaces between lables and textboxes such as in the customer tab, here, when you enetr the customer id input, it is not shown what you have eneterd until you reach a soecific number of letters
Look and Fed Proper spelling/grammar Look and Fed Message log entries scroll and display the most recent entry Justification as appropriate Message log entries scroll and display the most recent entry Justification as appropriate (e.g. numbers are right justified) All tab stops correct Reybaard shortcuts as appropriate Rey		professionalism			
Lock and in external contents are prefixed correctly and named in clear business terms Mental and an appropriate of the contents are removed explanatory All comments used comments exist Comments All comments used consensity experiences of the single propertience of the single properties are named or openitors or where the code is not self-explanatory All contents are prefixed correctly and named in clear business terms Methods and properties are named or openitors All contents used to self-explanatory Linding convention Variables and parameters are prefixed correctly and named in clear business terms Methods and properties are named or openitors Variables and parameters are prefixed correctly and named in clear business terms Methods and properties are named correctly Variables and parameters are prefixed correctly and named in clear business terms Methods and properties are named correctly Variables and parameters are prefixed correctly and named in clear business terms Methods and properties are named correctly Variables and parameters are prefixed correctly and named in clear business terms Methods and properties are named correctly Variables and parameters are prefixed correctly and named in clear business terms Methods and properties are named correctly Variables and parameters are prefixed correctly and named in clear business terms Methods and properties are named correctly All colors (Section Comments could be comment terms) Methods and properties are named correctly All colors (Section Comments could be comment to the colors of the colors					
Understandable lool lips as appropriate Nesses log serries scrib and display the most recent entry Justification as appropriate (e.g. numbers are right justified) All tab stops correct. Keyboard shortcuts as appropriate (e.g. numbers are right justified) Cursor reset to fit user input or composition of the state of the scrib input or composition of the scrib input of t	Look and Feel				
most recent entry Justification as appropriate (e.g. numbers are optin justified) All tab stops correct Keyboard shortcust as appropriate Cursor reset of fix user input errors Form reset after valid input General Supportability (applicable to all problem sets) Official class template used in all files Header All comments used in all files Emplate & Fice and Comments used in all files Header Comments used in all files Internal Comments Section comments exist where longer blocks or open user of comments FirmMain and all other code files are named properly Unlehemst are prefixed correctly and named in clear business terms (seception: ements and under code files are named properly Unlehemst are prefixed correctly and named in clear business terms (seception: ements on to used in code, like many labeling over- Code Style White space used effectively Good logical blocks of code (e.g. local variables and particular) Good separation of Ul, Business Logic, and obat functions 4.5 4.5 4.5 4.5 4.5 4.5 4.5 4.					
Justification as appropriate (e.g. numbers are right justified) All tab stops correct Keyboard shortcuts as appropriate Keyboard shortcuts as appropriate			-	45	
right justified) All tab stops correct Keyboard shortcuts as appropriate Appropriate Accept/Cancel functionality Cursor reset to fix use injust errors Form reset after valid input General Supportability (papticable to old problem sets) Official class template used in all files Header Completed for all files (Ms. 3-15)			J	4.5	
All tab stops correct keyboard shortcuts as appropriate Keyboard shortcuts as appropriate Keyboard shortcuts as appropriate Cursor reset to fix user input errors Form reset after valid imput General Supportability (applicable to all problem sets) Offical class template used in all files Header All code is in the proper template sections Empty procedures are removed All comments used surbusiness terms Method comments sed store business terms Method comments sed where longer blocks of code would benefit from them Une-by-line code for very technical opporations or where the code is not self-explanatory Find statements closed with a comment FrmMain and all other code files are named properly UI elements are prefixed correctly and named in clear business terms (exception: elements not used in code, like many labels) Variables and parameters are prefixed correctly and named in clear business terms (become in the code					
Seyboard shortcuts as appropriate Appropriate Accept/Cancel functionality Curor reset to fix use input errors Form reset after valid input Form r					
Usability Appropriate Accept/Cancel functionality Cursor reset for fix user input errors Form reset after valid input General Supportability (papicoble to all problem sets) Official class template used in all files Header Completed for all flies (INS 3-15) Header All code is in the proper template sections Empty procedures are removed All comments use clear business terms Method comments exist Section comments exist where longer blocks of code would benefit from them Une by-line code for very technical opporations or where the code is not seff-explanatory end statements closed with a comment FromMain and all other code files are named properly Un elements are prefixed correctly and named in clear business terms (exception: elements not used in code, like many lables) Variables and parameters are prefixed correctly and named in clear business terms Methods and properties are named correctly - Initialized Serinterface - October Syle White space used effectively - Good obgrant blocks of code (e.g. local variables defined all together) - Good separation of Ul, Business Logic, and Data functions		·			
Cursor reset to fix user input errors Form reset after valid input General Supportability (applicable to all problem sets) Template & Header All code is in the proper template sections Empty procedures are removed All comments use clear business terms Method comments uses All code would benefit from them Internal Comments Internal Internal Comments Internal Comments Internal Comments Internal Internal Comments Internal	Usability				
Form reset after valid input General Supportability (applicable to all problem sets) Template & Header Template & Header completed for all files (LNs 3-15) All codes is in the proper template sections Empty procedures are removed All coments use clear business terms Method comments exist Section comments exis	,	- ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' '		i	
General Supportability (applicable to all problem sets) Official class template used in all files Header All code is in the proper template sections Empty procedures are removed All comments use clear business terms Method comments exist Section comments exist sets Section comments exist where longer blocks of code would benefit from them Uniterly line code for very technical operations or where the code is not self-esplanatory End statements closed with a comment FrimMain and all other code files are named properly UI elements are prefixed correctly and named in clear business terms Membrads and parameters are prefixed correctly and named in clear business terms Methods and properties are named correctly InitializeUserInterface InitializeUserInterface JointalizeUserInterface Jointali					
Official class template used in all files Header completed for all files (LNs 3-15) All code is in the proper template sections Empty procedures are removed All comments use clear business terms Method comments exist Section comments exist where longer blocks of code would benefit from them Une-by line code for very technical operations or where the code is not self-explanatory End statements closed with a comment FrimMain and all other code files are named properly Ul elements are prefixed correctly and named in clear business terms (exception: lements not used in code, like many lables) Variables and parameters are prefixed correctly and named in clear business terms Methods and properties are named correctly initialize Userinterface Initialize Userinterface Jinitialize Userinterface Jinitialize Userinterface Jinitialize Userinterface Jostring private and public override in classes White space used effectively Good separation of Ui, Business Logic, and Data functions Source Style White space used effectively Good separation of Ui, Business Logic, and Data functions	Gonoral Supp	·			
Header completed for all files (LNs 3-15)	delleral suppl				
Template & Header		·	5		
All comments use clear business terms Method comments exist Section comments exist where longer blocks of code would benefit from them Une-by-line code for very technical operations or where the code is not self-explanatory End statements closed with a comment frmMain and all other code files are named properly UI elements are prefixed correctly and named in clear business terms (exception: elements not used in code, like many lables) Variables and parameters are prefixed correctly Variables and parameters are prefixed correctly and named in clear business terms Methods and properties are named correctly initialize_UserInterface	Template & Header				
Method comments exist Section comments exist where longer blocks of code would benefit from them Line-by-line code for very technical operations or where the code is not self-explanatory End statements closed with a comment FrimMain and all other code files are named properly UI elements are prefixed correctly and named in clear business terms (exception: elements not used in code, like many lables) Variables and parameters are prefixed correctly and named in clear business terms Methods and properties are named correctly — initializeUserInterface — initializeUserInterface — initializeUserInterface — initializeUserInterface — initializeUserInterface — finitializeUserInterface — food separation of UI, Business Logic, and Data functions Code Style Good logical blocks of code (e.g. local variables defined all together) Good separation of UI, Business Logic, and Data functions		Empty procedures are removed			
Section comments exist where longer blocks of code would benefit from them		All comments use clear business terms			
Internal Comments Comments Inine-by-line code for very technical operations or where the code is not self-explanatory End statements closed with a comment FrrmMain and all other code files are named properly UI elements are prefixed correctly and named in clear business terms (exception: elements not used in code, like many lables) Variables and parameters are prefixed correctly and named in clear business terms Methods and properties are named correctly					
Comments Line-by-line code for very technical operations or where the code is not self-explanatory End statements closed with a comment FrmMain and all other code files are named properly UI elements are prefixed correctly and named in clear business terms (exception: elements not used in code, like many lables) Variables and parameters are prefixed correctly and named in clear business terms Methods and properties are named correctly LinitializeUserinterface InitializeBusinessLogic ToString private and public override in classes White space used effectively Good logical blocks of code (e.g. local variables defined all together) Good separation of UI, Business Logic, and Data functions		Section comments exist where longer blocks		5	
operations or where the code is not self- explanatory End statements closed with a comment FrmMain and all other code files are named properly UI elements are prefixed correctly and named in clear business terms (exception: elements not used in code, like many lables) Variables and parameters are prefixed correctly and named in clear business terms Methods and properties are named correctlyinitializeUserInterfaceinitializeBusinessLogic ToString private and public override in classes White space used effectively Good logical blocks of code (e.g. local variables defined all together) Good separation of UI, Business Logic, and Data functions	Internal	of code would benefit from them			
explanatory End statements closed with a comment FrmMain and all other code files are named properly Ul elements are prefixed correctly and named in clear business terms (exception: elements not used in code, like many lables) Variables and parameters are prefixed correctly and named in clear business terms Methods and properties are named correctly initializeUserInterface initializeBusinessLogic ToString private and public override in classes White space used effectively Good logical blocks of code (e.g. local variables defined all together) Good separation of UI, Business Logic, and Data functions	Comments	Line-by-line code for very technical			
End statements closed with a comment FrmMain and all other code files are named properly UI elements are prefixed correctly and named in clear business terms (exception: elements not used in code, like many lables) Variables and parameters are prefixed correctly and named in clear business terms Methods and properties are named correctly LinitializeUserInterface LinitializeBusinessLogic ToString private and public override in classes White space used effectively Good logical blocks of code (e.g. local variables defined all together) Good separation of UI, Business Logic, and Data functions					
FrmMain and all other code files are named properly UI elements are prefixed correctly and named in clear business terms (exception: elements not used in code, like many lables) Variables and parameters are prefixed correctly and named in clear business terms Wethods and properties are named correctly initializeUserInterfaceinitializeBusinessLogic ToString private and public override in classes White space used effectively Good logical blocks of code (e.g. local variables defined all together) Good separation of UI, Business Logic, and Data functions					
Properly UI elements are prefixed correctly and named in clear business terms (exception: elements not used in code, like many lables) Variables and parameters are prefixed correctly and named in clear business terms Methods and properties are named correctly initializeUserInterfaceinitializeBusinessLogic ToString private and public override in classes White space used effectively Good logical blocks of code (e.g. local variables defined all together) Good separation of UI, Business Logic, and Data functions					
Naming convention Naming convention Variables and parameters are prefixed correctly and named in clear business terms (exception: elements not used in code, like many lables) Variables and parameters are prefixed correctly and named in clear business terms Methods and properties are named correctly initializeUserInterface initializeBusinessLogic ToString private and public override in classes White space used effectively Good logical blocks of code (e.g. local variables defined all together) Good separation of UI, Business Logic, and Data functions	Naming convention				
named in clear business terms (exception: elements not used in code, like many lables) Variables and parameters are prefixed correctly and named in clear business terms Methods and properties are named correctlyinitializeUserInterfaceinitializeBusinessLogic ToString private and public override in classes White space used effectively Good logical blocks of code (e.g. local variables defined all together) Good separation of UI, Business Logic, and Data functions					
Naming elements not used in code, like many lables) Variables and parameters are prefixed correctly and named in clear business terms Methods and properties are named correctly initializeUserInterfaceinitializeBusinessLogic ToString private and public override in classes White space used effectively Good logical blocks of code (e.g. local variables defined all together) Good separation of UI, Business Logic, and Data functions					
Convention Variables and parameters are prefixed correctly and named in clear business terms Methods and properties are named correctly initializeUserInterfaceinitializeBusinessLogic ToString private and public override in classes White space used effectively Good logical blocks of code (e.g. local variables defined all together) Good separation of UI, Business Logic, and Data functions		1 1			
Variables and parameters are prefixed correctly and named in clear business terms Methods and properties are named correctly initializeUserInterfaceinitializeBusinessLogic Methods ToString private and public override in classes White space used effectively Good logical blocks of code (e.g. local variables defined all together) Good separation of UI, Business Logic, and Data functions		ciements not used in code, like many lables)			
initializeUserInterfaceinitializeBusinessLogic ToString private and public override in classes White space used effectively Good logical blocks of code (e.g. local variables defined all together) Good separation of UI, Business Logic, and Data functions					
RequiredinitializeBusinessLogic		Methods and properties are named correctly			
RequiredinitializeBusinessLogic	Required Methods Code Style	initializeUserInterface			
Methods ToString private and public override in classes White space used effectively Good logical blocks of code (e.g. local variables defined all together) Good separation of UI, Business Logic, and Data functions		_			
Good logical blocks of code (e.g. local variables defined all together) Good separation of UI, Business Logic, and Data functions					
Good logical blocks of code (e.g. local variables defined all together) Good separation of UI, Business Logic, and Data functions		White space used offestively			
Code Style variables defined all together) Good separation of UI, Business Logic, and Data functions					
Good separation of UI, Business Logic, and Data functions					
Data functions					
nequired that open i uncalonality (draued as the project is running)	Required End	User Functionality (Graded as the project is ru	nning)		

RUN TIME	Data entry validated to prevent run time errors "Add/Create" buttons create correct output in Transaction Log "Test Data" button creates correct output in Transaction Log Exit button works	10	10	Summary/dashboard does not update correctly, such as the number/count of added objects adding a new object such as a customer is not updated
Required Cod	de Elements (Graded as a code review)			
FrmMain	Module level attribute and private property for ThemePark Proper variable definition / scope Click events for Create buttons Pass data to business logic classes Populate Transaction Log Click event for "Test Data" button using hardcoded data	8	7	1 in _runSystemTest(), you should only use the module ThemePark attribute (one object) to create/add the required objects such as customer and feature, this ThemePark object should create objects and track the current number of created objects, in one module ThemePark object, that is why it is defined as a module attribute in the FrmMain class
ThemePark	Attributes Public Properties Private Properties Create Procs for each Class Private/Public ToString	10	10	
Classes	Following Classes exist: ThemePark Customer Passbook Feature PassbookFeature UsedFeature For each class (refer to Class Diagrams): Attributes Special Constructor Public Properties Private/Public ToString	12	12	
	back and TOTAL			
Sample Rubric		50	48.5	