

Financial Management Software

Group Number Twenty Six

Date Submitted: November Twenty-Sixth, Two Thousand Two

Faculty Technical Advisor: Dr. Hong Man

John Comas

Steven Puzio

Pledge: "I pledge my honor that I have abided by the Stevens Honor System."

Table of Contents

I. Abstract	Page 1
II. Project Introduction	Page 2
III. Project Plan	Pages 3-5
IV. Prototype (Module 1)	Pages 6-7
V. Additional Features	Page 8
VI. Design Approaches	Page 9
VII. Financial Budget	Page 10
VIII. Gantt Charts	Pages 11-18
IX. Conclusion	Page 19
X. Appendix I	Pages 20-23
XI. Appendix II	Page 24

John Comas

Steven Puzio

CPE423: Engineering Design VII

Senior Design Final Project Report

Pledge: "I pledge my honor that I have abided by the Stevens Honor System."

Abstract:

This project involved the creation of a set of software tools designed to assist a person with managing their finances. A set of software tools was created component by component as separate small programs grouped together by a menu system which allows the user to switch between programs with ease. The software was written using Metrowerks CodeWarrior using the C++ development environment. Some of the tools which included in the package are loan calculators, income tax estimators, credit card interest calculators. Also, the software package includes a programming suite which will dynamically calculate an individual's adjusted gross income (AGI) and estimated capital gains tax and quarterly income tax if one sold current equity holdings based on current stock prices.

The project is approximately 50% complete at this stage. The work on module one (loan calculator) is complete and work on module two (credit card comparator) has been basically completed.

Project Introduction:

This project consists of the creation of software tools in an integrated development environment (IDE) namely Metrowerks CodeWarrior utilizing the C++ programming language. Since both group members are proficient in C++, there have been no technical difficulties up to this point. Since we do not require any physical parts or miscellaneous equipment, other than a desktop computer, the group feels strongly that the technical goals will be achieved. No problems have been encountered in the programming of module one and two at this time.

In this type of programming project, certain aspects of any software program can take longer to complete than originally estimated. Many times large companies such as Microsoft plan to have a software suite released by a certain date, but alas have to delay it due to program 'bugs' or additions which they wish to make to the suite. The group has decided that to promise a certain amount at the start and add to the suite later if time permits. In this manner, the components promised are being delivered on schedule and any additions will simply be 'icing on the cake.'

As noted previously, this project requires no special equipment or parts other than a personal computer. Since the group members are not being compensated for their time or programming knowledge, there are really no costs involved with the project. The only possible costs that are involved in this project are electricity to power the computers the group uses, wear & tear on the machines, paper for the required reports, and toner to print the reports on the paper. Since these costs are relatively small, they are being absorbed by the design group members and the project has not run out of money or become over budget.

Project Plan:

The project the group is working on is, as expected, approximately 50% complete. The group anticipates moving very far along in the project during the intersession break. Programming has been completed on the first two modules of the software, and the project should be at least 75% complete if not fully completed by the beginning of the Spring 2003 semester. As stated previously, the software suite as this time will consist of four main smaller programs. At present, the delivered programs will consist of the following:

- 1.) A loan calculator which will give a user the current payoff amount of a loan, the interest paid YTD, and the monthly payments. This module is complete
- 2.) A credit card calculator which will allow the user to see the interest being charged on their credit cards based on the average daily balance method with new charges and current payments made. This module is complete.
- 3.) An income tax estimator which will, based on employment information entered by the user, project the estimated quarterly tax payments for a future fiscal year. Work on this module is to be complete during the intersession break.
- 4.) A programming suite which will dynamically calculate an individual's adjusted gross income (AGI) and estimated capital gains tax and quarterly income tax if one sold current equity holdings based on current stock prices. The program will in effect create a virtual 1099 form that the person can easily view without having to wait for the EOY report from their financial institution. Work on this module is expected to take place, if time permits,

during the intersession break or the Spring 2003 semester. This module is the most complex and will take the most time to complete. The group's advisor, Prof. Man, has suggested that we utilize the XML programming language in order to complete this module. Since the group is not readily proficient in XML, there will be a steep learning curve with respect to this module if XML is implemented. Therefore, if XML is utilized in this module, it may take at least half of the Spring 2003 semester to be completed.

At this point, it is clear to see that the software is as powerful as it is was projected to be. The first two modules which have been completed are very useful, and could probably sold on their own. Since the project is so far quite successful, the group feels that it could in fact market the suite to a software company which can sell it to consumers. This software is being written in C++ using the Metrowerks CodeWarrior IDE in a standard C++ console. The software is thus portable to any platform such as the Macintosh OS, UNIX, or MS Windows. Application Program Interfaces (API's) can be utilized to turn this software into a platform specific suite.

At this time, there have been no problems arising with the development of this software. If programming questions come up during the ongoing software construction, they can be addressed at that time. The group feels very strongly that it can complete the promised programs with full functionality by the required time.

The average taxpayer usually starts working on his or her income tax after that tax year has already ended and rushes to complete the work by April 15th so that they do not have to go on extension. Going on extension can sometimes leave a taxpayer at a

disadvantage because they may have to compensate the IRS with underpayment penalties if they do not pay their taxes in full by April 15th. The suite allows a user to dynamically calculate their AGI and income taxes so that he or she will know exactly how much tax to pay on their quarterly taxes and avoid underpayments. The group feels strongly this feature makes consumers want to purchase the software and its ease of use makes it very popular in the marketplace.

Our software suite requires no special equipment or parts other than a personal computer. Since this software is in an 'alpha' stage of development at this point, no additional programmers are required in order to bring the project to completion. Each of the group members has his own computers and integrated development environments. Therefore, no special equipment or software had to be outright purchased in order to complete the project. Since the group members are not being compensated for their time or programming knowledge, the project has a zero cost basis. If all possible costs were to be factored in, we the electricity required to power the computers the group uses, wear & tear on the group's computers & keyboards, paper for the required reports, and toner to print the reports on the paper could be included. Since these costs are relatively small, they are not being budgeted as project expenses and have been absorbed by the design group members.

Prototype:

The current prototyped program provides all of the functionality originally specified. Since at the time of this report's construction, the credit card comparator module is just nearing completion and bug testing, the loan calculator program will be detailed on its own. Contained in appendix (I) is contained the source code of the core of the loan calculator prototyped program. The core of the program provides the following functionality:

- a.) The user enters a loan amount in U.S. dollars.
- b.) The interest rate in whole numbers e.g. The user enters 6.02 for 6.02%.
- c.) The length of the loan in months. e.g. The user enter 120 for 12 years or 120 months.
- d.) The program immediately outputs the monthly payment required in order to pay off the loan with the specified parameters. The total interest which will be paid out over the life of the loan.
- e.) The core program also provides an inflation adjusted interest cost which economists use to estimate the 'true' value that the money which was paid in interest would have had if it was saved by the payee.

The aforementioned is the core of the loan calculator program. Further amortization features will be added as time permits, and the program will be expanded next year. Contained in appendix (II) is the output of the compiled loan calculator program. The example used is a \$75,000 loan at 6.02% for 10 years. The program clearly shows that the monthly loan payment is \$833.41 per month, and the interest which will be paid to the

lender over the life of the loan is \$25,008.60. If a 3% inflation number is inputted into the program, the user can see that the actual inflation adjusted interest cost is \$33,745.70.

In order to calculate this data, the program uses the formulas $a = [P((1 + r)^n)r]/[(1 + r)^n - 1]$ and $FV = P(1 + r/n)^Y$ in order to calculate respectively the interest paid and inflation adjusted interest cost. These formulas are very easily calculable by the C++ language, and thus the program is highly efficient and performs very well in real life tests. Approximately 100 tests were performed on the loan calculator to assure that it was outputting correct data. Since the aforementioned formulas are sound and accurate, there are no errors seen in the functionality of the program. Performance is as expected in the program. Even the slowest of computers can calculate formulas such as these with relative ease. Thus any computer can run this module and the user can experience a high level of performance.

Additional Features:

If time permits, the group may add several features to the software project. One interesting feature being considered is the creation of a set of prefabricated income tax form sheets which will allow people to not only see their income tax forms on the computer screen, but watch them dynamically change with modifications they make to their income or stock portfolios. This will require a significant amount of time to complete so the group is unsure if this can be completed in time with the other promised elements.

This software will run under pretty much any type of operating system and on most personal computers. Newer or more advanced hardware will speed up the software suite, but will not change the functionality of the software.

Design Approaches:

In order to complete this project, the group has decided that the development will take a 'see-as-we-go' approach. The software has the four clear, promised deliverables which were detailed earlier. Two of these deliverables are now complete and on schedule. Since software programs typically take longer to complete than estimated, the group will focus on completing the next two elements before any more is taken on.

Additional functions will be added later as time permits. Software programming does not usually take very long to outline and key in. The very time consuming aspect to programming is the bug fixing and refining stage. As is well known by professional programmers, it can sometimes take a week to write a large program and a month to fix all of the bugs.

Financial Budget:

The most costly aspect of writing software is paying programmers to construct it. In industry, this makes up the bulk of the cost of the program. The only materials required for the construction of software are computers, an integrated development environment, possibly some text references, and CD writers & blank CD's to burn the software. There are no moving parts or other pieces required in programming. Computer equipment is required in order to test the product, but this is can be the same hardware used to write the software. No other special computers are required. If the code of a software suite is to be printed out using a printer, the documentation costs of doing so can be significant. Programs consist of thousands and sometimes millions of lines of code. Printing all of this out can take hundreds of pieces of paper and many printers. It is estimated that the cost of printing out a single page is 7 cents. Travel, hotel accommodations, and meals for the engineering staff who travel periodically customers to discuss the project status can be significant. It is usually necessary to rent them an upscale room in a hotel such as the Waldorf Astoria or The Plaza Hotel and wine & dine them at restaurants such as Tavern on the Green. For that reason, these costs can be quite high especially if they eat as much as the group leader does. This can make up about 5% of the cost of every product sold. As with any software tool, technical support is necessary and must be provided to the consumer. However, as many companies are doing today, this support is not free, and its operation will be paid for by the consumer directly. Rent and utilities will make up at least another 5% of the total product cost. A profit will be created because the product will be priced high enough to allow for one to occur.

Conclusion:

The project at this point has so far been a very successful implementation of our objectives. The group feels that the elements of this software suite will be highly desired by consumers and will prove to be an essential part of a person's software on their personal computers. Since the programming stage is approximately 50% complete, it is clear the project is implementable and successful. Once the four main modules are complete, the group can work on other additions to the suite.

Appendix (I):

SOURCE CODE OF THE LOAN CALCULATOR (MODULE 1):

```
//CPE423: Senior Design
```

```
//Module 1: Loan Calculator
```

```
#include <iostream.h>
```

```
#include <math.h>
```

```
float LoanAmount, InterestRate, BalanceA, BalanceB, Payment, DecimalInterestRate,  
LoanPayment;
```

```
int NumberOfMonths, i, MonthlyInterest;
```

```
float MonthlyPayment, InterestPaid;
```

```
int main ()
```

```
{
```

```
cout<<"Please enter the loan amount: ";
```

```
cin>>LoanAmount;
```

```
cout<<endl<<endl<<"Please enter the interest rate: ";
```

```
cin>>InterestRate;
```

```
cout<<endl<<endl<<"Please enter the loan duration in months: ";
```

```
cin>>NumberOfMonths;
```

```
cout<<endl<<endl<<endl;
```

```
DecimalInterestRate=(InterestRate/100)/12;
```

```
//Formula used:  $a = [P((1 + r)^n)r]/[(1 + r)^n - 1]$ 
```

```
BalanceA=LoanAmount*(pow((1+DecimalInterestRate),NumberOfMonths))*DecimalInterestRate;
```

```
BalanceB=(pow(1+DecimalInterestRate,NumberOfMonths))-1;
```

```
MonthlyPayment=BalanceA/BalanceB;
```

```
cout<< "Monthly Payment on the loan is $"<< MonthlyPayment<< " for  
"<<NumberOfMonths<<" months."<<endl<<endl;
```

```
InterestPaid=(MonthlyPayment*NumberOfMonths)-LoanAmount;
```

```
cout<<"For a "<<NumberOfMonths<<" month amortization, the interest paid is  
$"<<InterestPaid<<". "<<endl<<endl;
```

```
float InflationAdjIntCost, RateOfInflation, WholeInflationRate;
```

```
cout<<"Please enter an estimated yearly rate of inflation: ";
```

```
cin>>WholeInflationRate;
```

```
cout<<endl<<endl;
```

```
RateOfInflation=WholeInflationRate/100;
```

```
//Formula used:  $FV = P(1 + r/n)^Y$ 
```

```
InflationAdjIntCost=InterestPaid*(pow(1+(RateOfInflation/12),NumberOfMonths));
```

```
cout<<"Assuming a "<<WholeInflationRate<<"% rate of inflation, the inflation adjusted  
interest cost is $"<<InflationAdjIntCost;
```

```
return 0;
```

```
}
```

Appendix (II):

OUTPUT OF THE LOAN CALCULATOR (MODULE 1):

Please enter the loan amount: 75000

Please enter the interest rate: 6.02

Please enter the loan duration in months: 120

Monthly Payment on the loan is \$833.41 for 120 months.

For a 120 month amortization, the interest paid is \$25008.60.

Please enter an estimated yearly rate of inflation: 3

Assuming a 3% rate of inflation, the inflation adjusted interest cost is \$33745.70.