

Selected files

7 printable files

src/AmortizedLoan.cpp
src/AmortizedLoan.h
src/Loan.cpp
src/Loan.h
src/main.cpp
src/SimpleLoan.cPP
src/SimpleLoan.h

src/AmortizedLoan.cpp

```
1  /*
2  file: AmortizedLoan.cpp
3  Name: Elijah Heimsoth
4  Date: 4/11/2024
5  Description: This is the implementation file for the AmortizedLoan class.
6  It contains the implementation of the class functions.
7
8  FUNCTIONS:
9
10 // Default constructor
11 AmortizedLoan();
12 Receives: Nothing
13 Returns: Nothing
14 Description: This is the default constructor for the AmortizedLoan class.
15
16 // Parameterized constructor
17 AmortizedLoan(float aPrincipal, float aInterestRate, int aLoanLength);
18 Receives: aPrincipal, aInterestRate, aLoanLength
19 Returns: Nothing
20 Description: This is the parameterized constructor for the AmortizedLoan class.
21
22 // Destructor
23 virtual ~AmortizedLoan();
24 Receives: Nothing
25 Returns: Nothing
26 Description: This is the destructor for the AmortizedLoan class.
27
28 // Override the monthly payment function
29 float monthlyPayment() override;
30 Receives: Nothing
31 Returns: a float
32 Description: This function overrides the monthly payment function from the Loan class.
33 It calculates the monthly payment for an amortized loan.
34 */
35
36 #include "AmortizedLoan.h"
37 #include <cmath>
38
39 // Default constructor
40 AmortizedLoan::AmortizedLoan() {};
41
```

```

42 /* ^^^EQUIVALENT TO^^^:
43 AmortizedLoan::AmortizedLoan() {
44     principal = 0.0f;
45     interestRate = 0.0f;
46     loanLength = 0;
47     loanType = "";
48 }
49 */
50
51 // Parameterized constructor
52 AmortizedLoan::AmortizedLoan(float aPrincipal, float aInterestRate, int aLoanLength)
53     : Loan(aPrincipal, aInterestRate, aLoanLength, "Amortized Loan") {}
54
55 /* ^^^EQUIVALENT TO^^^:
56 AmortizedLoan::AmortizedLoan(float aPrincipal, float aInterestRate, int aLoanLength) {
57     principal = aPrincipal;
58     interestRate = aInterestRate;
59     loanLength = aLoanLength;
60     loanType = "Amortized Loan";
61 }
62 */
63
64 // Destructor
65 AmortizedLoan::~AmortizedLoan() {
66     // Empty
67 }
68
69 // Override the monthly payment function
70 float AmortizedLoan::monthlyPayment() {
71     // MONTHLY PAYMENT: (P * R * ((1+R)^L) ) / (((1+R)^L) - 1)
72     // P = principal, R = monthly interest rate, L = total months
73     float monthlyRate = interestRate / 1200.0f;
74     int totalMonths = loanLength * 12;
75     float monthlyPayment = (principal * monthlyRate * pow(1 + monthlyRate,
76     totalMonths)) / (pow(1 + monthlyRate, totalMonths) - 1);
77     return monthlyPayment;
78 }
```

src/AmortizedLoan.h

```

1 /*
2 file: AmortizedLoan.h
3 Name: Elijah Heimsoth
4 Date: 4/11/2024
5 Description: This is the header file for the AmortizedLoan class.
6 It contains the prototyping for the class.
7
8 FUNCTIONS:
9
10 // Default constructor
11 AmortizedLoan();
12 Receives: Nothing
13 Returns: Nothing
14 Description: This is the default constructor for the AmortizedLoan class.
15
```

```

16 // Parameterized constructor
17 AmortizedLoan(float aPrincipal, float aInterestRate, int aLoanLength);
18 Receives: aPrincipal, aInterestRate, aLoanLength
19 Returns: Nothing
20 Description: This is the parameterized constructor for the AmortizedLoan class.
21
22 // Destructor
23 virtual ~AmortizedLoan();
24 Receives: Nothing
25 Returns: Nothing
26 Description: This is the destructor for the AmortizedLoan class.
27
28 // Override the monthly payment function
29 float monthlyPayment() override;
30 Receives: Nothing
31 Returns: a float
32 Description: This function overrides the monthly payment function from the Loan class.
33 It calculates the monthly payment for an amortized loan.
34 */
35
36 #ifndef AMORTIZEDLOAN_H
37 #define AMORTIZEDLOAN_H
38
39 #include "Loan.h"
40
41 class AmortizedLoan : public Loan {
42 public:
43     // Default constructor
44     AmortizedLoan();
45
46     // Parameterized constructor
47     AmortizedLoan(float aPrincipal, float aInterestRate, int aLoanLength);
48
49     // Destructor
50     virtual ~AmortizedLoan();
51
52     // Override the monthly payment function
53     float monthlyPayment() override;
54 };
55
56 #endif

```

src/Loan.cpp

```

1 /*
2  file: Loan.cpp
3  Name: Elijah Heimsoth
4  Date: 4/11/2024
5  Description: This is the implementation file for the Loan class.
6  It contains the implementation of the class functions.
7
8  FUNCTIONS:
9
10 // Default constructor
11 Loan();

```

```
12 Receives: Nothing
13 Returns: Nothing
14 Description: This is the default constructor for the Loan class.
15
16 // Parameterized constructor
17 explicit Loan(float aPrincipal, float aInterestRate, int aLoanLength, std::string
18 aLoanType);
19 Receives: aPrincipal, aInterestRate, aLoanLength, aLoanType
20 Returns: Nothing
21 Description: This is the parameterized constructor for the Loan class.
22
23 // Destructor
24 virtual ~Loan();
25 Receives: Nothing
26 Returns: Nothing
27 Description: This is the destructor for the Loan class.
28
29 // Getters
30 float getPrincipal();
31 Receives: Nothing
32 Returns: a float, principal
33 Description: This function returns the principal of the loan.
34
35 float getInterestRate();
36 Receives: Nothing
37 Returns: a float, interestRate
38 Description: This function returns the annual interest rate of the loan.
39
40 float getLoanLength();
41 Receives: Nothing
42 Returns: an float, loanLength
43 Description: This function returns the length of the loan in years.
44
45 std::string getLoanType();
46 Receives: Nothing
47 Returns: a string, loanType
48 Description: This function returns the type of the loan.
49
50 // Setters
51 void setPrincipal(float thePrincipal);
52 Receives: a float, thePrincipal
53 Returns: Nothing
54 Description: This function sets the principal of the loan.
55
56 void setInterestRate(float theInterestRate);
57 Receives: a float, theInterestRate
58 Returns: Nothing
59 Description: This function sets the annual interest rate of the loan.
60
61 void setLoanLength(int theLoanLength);
62 Receives: an int, theLoanLength
63 Returns: Nothing
64 Description: This function sets the length of the loan in years.
65
66 void setLoanType(std::string theLoanType);
67 Receives: a string, theLoanType
```

```
67 Returns: Nothing
68 Description: This function sets the type of the loan.
69
70 // Base class functions
71 virtual float monthlyPayment() = 0; // Pure virtual function
72 Receives: Nothing
73 Returns: a float, monthlyPayment
74 Description: This function calculates the monthly payment of the loan.
75
76 void displayLoan();
77 Receives: Nothing
78 Returns: Nothing
79 Description: This function displays the loan information to the console.
80
81 // File I/O
82 void saveLoan();
83 Receives: Nothing
84 Returns: Nothing
85 Description: This function saves the loan information to a file.
86 */
87
88 #include "Loan.h"
89 #include <iostream>
90 #include <fstream>
91 #include <iomanip>
92 #include <string>
93
94 // Default constructor
95 Loan::Loan() {
96     principal = 0.0f;
97     interestRate = 0.0f;
98     loanLength = 0;
99     loanType = "";
100 }
101
102 // Parameterized constructor
103 Loan::Loan(float aPrincipal, float aInterestRate, int aLoanLength, std::string aLoanType) {
104     principal = aPrincipal;
105     interestRate = aInterestRate;
106     loanLength = aLoanLength;
107     loanType = aLoanType;
108 }
109
110 // Virtual destructor
111 Loan::~Loan() {
112     // Empty
113 }
114
115 // Getters
116 float Loan::getPrincipal() {
117     return principal;
118 }
119
120 float Loan::getInterestRate() {
121     return interestRate;
```

```
122 }
123
124 float Loan::getLoanLength() {
125     float loanLengthFloat = static_cast<float>(loanLength);
126     return loanLengthFloat;
127 }
128
129 std::string Loan::getLoanType() {
130     return loanType;
131 }
132
133 // Setters
134 void Loan::setPrincipal(float thePrincipal) {
135     principal = thePrincipal;
136 }
137
138 void Loan::setInterestRate(float theInterestRate) {
139     interestRate = theInterestRate;
140 }
141
142 void Loan::setLoanLength(int theLoanLength) {
143     loanLength = theLoanLength;
144 }
145
146 void Loan::setLoanType(std::string theLoanType) {
147     loanType = theLoanType;
148 }
149
150 // displayLoan: Outputs the loan information to the console
151 void Loan::displayLoan() {
152     std::cout << "Loan Overview" << std::endl;
153     std::cout << std::string(32, '=') << std::endl;
154     std::cout << std::setw(18) << std::left << "Loan Type:" << getLoanType() <<
155     std::endl;
156     std::cout << std::setw(18) << std::left << "Principal:" << getPrincipal() <<
157     std::endl;
158     std::cout << std::setw(18) << std::left << "Interest Rate:" << getInterestRate()
159     << "%" << std::endl;
160     std::cout << std::setw(18) << std::left << "Length in Years:" << getLoanLength()
161     << std::endl;
162     std::cout << std::setw(18) << std::left << "Monthly Payment:" << monthlyPayment()
163     << std::endl;
164 }
165
166 // saveLoan: Writes the loan information to a file
167 void Loan::saveLoan() {
168     // Directory and file name
169     const std::string directory = "data/";
170     const std::string filename = "loans.txt";
171
172     // Open file in output mode (overwrites existing file or creates a new one if it
173     // doesn't exist)
174     std::ofstream outFile(directory + filename, std::ios::out);
175
176     // Write the loan information to the file
177     outFile << getPrincipal() << " "
178         << getInterestRate() << " "
179         << getLoanLength() << " "
180         << monthlyPayment() << " "
181         << loanType << " "
182         << interestRate << "%"
183 }
```

```

174     << getLoanLength() << " "
175     << std::endl;
176
177     // Close the file
178     outFile.close();
179
180     /*
181     // Check if the file is open
182     if (outFile.is_open()) {
183         // Write the loan information to the file
184         outFile << getPrincipal() << " "
185             << getInterestRate() << " "
186             << getLoanLength() << " "
187             << std::endl;
188
189         // Close the file
190         outFile.close();
191     } else {
192         std::cerr << "Unable to open file for writing." << std::endl;
193     }
194     */
195 }
196

```

src/Loan.h

```

1  /*
2  file: Loan.h
3  Name: Elijah Heimsoth
4  Date: 4/11/2024
5  Description: This is the header file for the Loan class.
6  It contains the prototyping for the class.
7
8  FUNCTIONS:
9
10 // Default constructor
11 Loan();
12 Receives: Nothing
13 Returns: Nothing
14 Description: This is the default constructor for the Loan class.
15
16 // Parameterized constructor
17 explicit Loan(float aPrincipal, float aInterestRate, int aLoanLength, std::string
18 aLoanType);
19 Receives: aPrincipal, aInterestRate, aLoanLength, aLoanType
20 Returns: Nothing
21 Description: This is the parameterized constructor for the Loan class.
22
23 // Destructor
24 virtual ~Loan();
25 Receives: Nothing
26 Returns: Nothing
27 Description: This is the destructor for the Loan class.
28 // Getters

```

```
29 float getPrincipal();
30 Receives: Nothing
31 Returns: a float, principal
32 Description: This function returns the principal of the loan.
33
34 float getInterestRate();
35 Receives: Nothing
36 Returns: a float, interestRate
37 Description: This function returns the annual interest rate of the loan.
38
39 float getLoanLength();
40 Receives: Nothing
41 Returns: an float, loanLength
42 Description: This function returns the length of the loan in years.
43
44 std::string getLoanType();
45 Receives: Nothing
46 Returns: a string, loanType
47 Description: This function returns the type of the loan.
48
49 // Setters
50 void setPrincipal(float thePrincipal);
51 Receives: a float, thePrincipal
52 Returns: Nothing
53 Description: This function sets the principal of the loan.
54
55 void setInterestRate(float theInterestRate);
56 Receives: a float, theInterestRate
57 Returns: Nothing
58 Description: This function sets the annual interest rate of the loan.
59
60 void setLoanLength(int theLoanLength);
61 Receives: an int, theLoanLength
62 Returns: Nothing
63 Description: This function sets the length of the loan in years.
64
65 void setLoanType(std::string theLoanType);
66 Receives: a string, theLoanType
67 Returns: Nothing
68 Description: This function sets the type of the loan.
69
70 // Base class functions
71 virtual float monthlyPayment() = 0; // Pure virtual function
72 Receives: Nothing
73 Returns: a float, monthlyPayment
74 Description: This function calculates the monthly payment of the loan.
75
76 void displayLoan();
77 Receives: Nothing
78 Returns: Nothing
79 Description: This function displays the loan information to the console.
80
81 // File I/O
82 void saveLoan();
83 Receives: Nothing
84 Returns: Nothing
```

```

85 | Description: This function saves the loan information to a file.
86 |
87 |
88 | #ifndef LOAN_H
89 | #define LOAN_H
90 |
91 | #include <iostream>
92 |
93 | class Loan {
94 | protected:
95 |     float principal;
96 |     float interestRate;
97 |     int loanLength;
98 |     std::string loanType;
99 |
100 | public:
101 |     // Default constructor
102 |     Loan();
103 |
104 |     // Parameterized constructor
105 |     explicit Loan(float aPrincipal, float aInterestRate, int aLoanLength, std::string
106 |     aLoanType);
107 |
108 |     // Destructor
109 |     virtual ~Loan();
110 |
111 |     // Getters
112 |     float getPrincipal();
113 |     float getInterestRate();
114 |     float getLoanLength();
115 |     std::string getLoanType();
116 |
117 |     // Setters
118 |     void setPrincipal(float thePrincipal);
119 |     void setInterestRate(float theInterestRate);
120 |     void setLoanLength(int theLoanLength);
121 |     void setLoanType(std::string theLoanType);
122 |
123 |     // Base class functions
124 |     virtual float monthlyPayment() = 0; // Pure virtual function
125 |     void displayLoan();
126 |
127 |     // File I/O
128 |     void saveLoan();
129 |
130 | #endif
131 |

```

src/main.cpp

```

1 | /*
2 | file: main.cpp
3 | Name: Elijah Heimsoth
4 | Date: 4/11/2024

```

```
5 | Description: This is the main file for the Loan program.  
6 | It contains the main function that tests the Loan classes.  
7 |  
8 | FUNCTIONS:  
9 |  
10 | // Main function  
11 | int main();  
12 | Receives: Nothing  
13 | Returns: 0  
14 | Description: This is the main function for the Loan program.  
15 | */  
16 |  
17 |#include "Loan.h"  
18 |#include "SimpleLoan.h"  
19 |#include "AmortizedLoan.h"  
20 |#include <iostream>  
21 |  
22 |int main() {  
23 |    // Create an empty simple loan  
24 |    SimpleLoan emptySimpleLoan;  
25 |    std::cout << "Empty Simple Loan:" << std::endl;  
26 |    emptySimpleLoan.displayLoan();  
27 |    std::cout << std::endl;  
28 |  
29 |    // Create a simple loan with parameters  
30 |    // simpleLoan(float aPrincipal, float aInterestRate, int aLoanLength)  
31 |    SimpleLoan testSimpleLoan(1000, 5, 4);  
32 |    std::cout << "Simple Loan:" << std::endl;  
33 |    testSimpleLoan.displayLoan();  
34 |    std::cout << std::endl;  
35 |  
36 |    // Create an empty Amortized Loan  
37 |    AmortizedLoan emptyAmortizedLoan;  
38 |    std::cout << "Empty Amortized Loan:" << std::endl;  
39 |    emptyAmortizedLoan.displayLoan();  
40 |    std::cout << std::endl;  
41 |  
42 |    // Create an Amortized Loan with parameters  
43 |    //amortizedLoan(float aPrincipal, float aInterestRate, int aLoanLength)  
44 |    AmortizedLoan testAmortizedLoan(1000, 5, 5);  
45 |    std::cout << "Amortized Loan:" << std::endl;  
46 |    testAmortizedLoan.displayLoan();  
47 |    std::cout << std::endl;  
48 |  
49 |    // Test saveLoan function  
50 |    testSimpleLoan.saveLoan();  
51 |    testAmortizedLoan.saveLoan();  
52 |  
53 |    return 0;  
54 |}  
55 |
```

src/SimpleLoan.cpp

1 | /*

```
2 file: SimpleLoan.cpp
3 Name: Elijah Heimsoth
4 Date: 4/11/2024
5 Description: This is the implementation file for the SimpleLoan class.
6 It contains the implementation of the class functions.
7
8 FUNCTIONS:
9
10 // Default constructor
11 SimpleLoan();
12 Receives: Nothing
13 Returns: Nothing
14 Description: This is the default constructor for the SimpleLoan class.
15
16 // Parameterized constructor
17 explicit SimpleLoan(float aPrincipal, float aInterestRate, int aLoanLength);
18 Receives: aPrincipal, aInterestRate, aLoanLength
19 Returns: Nothing
20 Description: This is the parameterized constructor for the SimpleLoan class.
21
22 // Destructor
23 virtual ~SimpleLoan();
24 Receives: Nothing
25 Returns: Nothing
26 Description: This is the destructor for the SimpleLoan class.
27
28 // Override the monthly payment function
29 float monthlyPayment() override;
30 Receives: Nothing
31 Returns: a float
32 Description: This function overrides the monthly payment function from the Loan class.
33 It calculates the monthly payment for a simple loan.
34 */
35
36 #include "SimpleLoan.h"
37
38 // Default constructor
39 SimpleLoan::SimpleLoan() {};
40
41 /* ^^^EQUIVALENT TO^^^:
42 SimpleLoan::SimpleLoan() {
43     principal = 0.0f;
44     interestRate = 0.0f;
45     loanLength = 0;
46     loanType = "";
47 }
48 */
49
50 // Parameterized constructor
51 SimpleLoan::SimpleLoan(float aPrincipal, float aInterestRate, int aLoanLength)
52     : Loan(aPrincipal, aInterestRate, aLoanLength, "Simple Loan") {}
53
54 /* ^^^EQUIVALENT TO^^^:
55 SimpleLoan::SimpleLoan(float aPrincipal, float aInterestRate, int aLoanLength) {
56     principal = aPrincipal;
57     interestRate = aInterestRate;
```

```

58     loanLength = aLoanLength;
59     loanType = "Simple Loan";
60 }
61 */
62
63 // Destructor
64 SimpleLoan::~SimpleLoan() {
65     // Empty
66 }
67
68 // Override the monthly payment function
69 float SimpleLoan::monthlyPayment() {
70     // MONTHLY PAYMENT: (P*(R*L + 1)) / L
71     // P = principal, R = monthly interest rate, L = total months
72     float monthlyRate = interestRate / 1200.0f;
73     int totalMonths = loanLength * 12;
74     float monthlyPayment = (principal * (monthlyRate * totalMonths + 1)) /
totalMonths;
75     return monthlyPayment;
76 }
77
78
79
80
81
82
83
84
85
86
87
88
89
90

```

src/SimpleLoan.h

```

1 /*
2 file: SimpleLoan.h
3 Name: Elijah Heimsoth
4 Date: 4/11/2024
5 Description: This is the header file for the SimpleLoan class.
6 It contains the prototyping for the class.
7
8 FUNCTIONS:
9
10 // Default constructor
11 SimpleLoan();
12 Receives: Nothing
13 Returns: Nothing
14 Description: This is the default constructor for the SimpleLoan class.
15
16 // Parameterized constructor
17 explicit SimpleLoan(float aPrincipal, float aInterestRate, int aLoanLength);
18 Receives: aPrincipal, aInterestRate, aLoanLength

```

```
19 Returns: Nothing
20 Description: This is the parameterized constructor for the SimpleLoan class.
21
22 // Destructor
23 virtual ~SimpleLoan();
24 Receives: Nothing
25 Returns: Nothing
26 Description: This is the destructor for the SimpleLoan class.
27
28 // Override the monthly payment function
29 float monthlyPayment() override;
30 Receives: Nothing
31 Returns: a float
32 Description: This function overrides the monthly payment function from the Loan class.
33 It calculates the monthly payment for a simple loan.
34 */
35
36 #ifndef SIMPLELOAN_H
37 #define SIMPLELOAN_H
38
39 #include "Loan.h"
40
41 class SimpleLoan : public Loan {
42 public:
43     // Default constructor
44     SimpleLoan();
45
46     // Parameterized constructor
47     explicit SimpleLoan(float aPrincipal, float aInterestRate, int aLoanLength);
48
49     // Virtual destructor
50     virtual ~SimpleLoan();
51
52     // Override the monthly payment function
53     float monthlyPayment() override;
54 };
55
56 #endif
```