“Heaven’s light is our guide”

**Rajshahi University of Engineering & Technology**

**Department of Computer Science & Engineering**

**Course Title: Object Oriented Programming Sessional**

**Course No: CSE 1204**

Submitted To:

***Rizoan Toufiq***

*Assistant Professor,*

*Dept. of CSE,*

*RUET.*

Submitted By:

***Saifur Rahman***

*Roll No: 1703018*

*Section: ’17-A*

*Class: 1st year (Even Semester)*

***Date of Submission: 14 Nov, 2018***

**INDEX**

|  |  |  |
| --- | --- | --- |
| **Program No.** | **Topic** | **Name** |
| **1.** | **Structure** | **A bank certificate of deposit.** |
| **2.** | **Class & Object** | **Class with a member function.** |
| **3.** | **Class & Object** | **The bank account class.** |
| **4.** | **Class & Object** | **Constructor and Destructor.** |
| **5.** | **Class & Object** | **Time interval of a program using class.** |
| **6.** | **Class & Object** | **Dynamic memory allocation of a string in class.** |
| **7.** | **Class & Object** | **Overloading of a constructor** |
| **8.** | **Class & Object** | **Object pointer** |

**1. Structure for a bank certificate of deposit.**

#include <iostream>

using namespace std;

struct CdAc{

double bal;

double intrate;

int term;

};

void getData(CdAc &theAc);

int main( )

{

CdAc acc;

getData(acc);

double ratefrac, interest;

ratefrac = acc.intrate / 100.00;

interest = acc.bal \* ratefrac \* (acc.term / 12.0);

acc.bal = acc.bal + interest;

cout.setf(ios::fixed);

cout.setf(ios::showpoint);

cout.precision(2);

cout << "When your CD matures in "

<< acc.term << " months,\n"

<< "it will have a balance of $"

<< acc.bal << endl;

}

void getData(CdAc &theAc){

cout << "Enter account balance: $ ";

cin >> theAc.bal;

cout << "Enter account interest rate: ";

cin >> theAc.intrate;

cout << "Enter the number of months until maturity\n"

<< "(12 months or fewer)\n";

cin >> theAc.term;

}

**2. Class with a member function.**

#include <iostream>

using namespace std;

class DayOfYear{

public:

void output( );

int m;

int d;

};

int main( )

{

DayOfYear today, bday;

cout << "Enter today's date:\n";

cout << "Enter month as a number: ";

cin >> today.m;

cout << "Enter the day of the month: ";

cin >> today.d;

cout << "Enter your birthday:\n";

cout << "Enter month as a number: ";

cin >> bday.m;

cout << "Enter the day of the month: ";

cin >> bday.d;

cout << "Today's date is ";

today.output( );

cout << "Your birthday is ";

bday.output( );

if (today.m == bday.m && today.d == bday.d)

cout << "Happy Birthday!\n";

else

cout << "Not your birthday!\n";

return 0;

}

void DayOfYear::output( ){

cout << "month = " << m

<< ", day = " << d << endl;

}

**3. The bank account class.**

#include <iostream>

using namespace std;

class BankAccount{

public:

void set(int d, int c, double r);

void set(int d, double r);

void update( );

double getBal( );

double getRate( );

void output(ostream& outs);

private:

double bal;

double intRate;

double frac(double percent);

};

int main( )

{

BankAccount a1, a2;

int d1, d2, c1, c2;

double r1, r2;

cout << "Start of Test:\n";

cout << "Enter amount of dollars: ";

cin >> d1;

cout << "Enter amount of cents: ";

cin >> c1;

cout << "Enter rate: ";

cin >> r1;

a1.set(d1, c1, r1);

cout << "account1 initial statement:\n";

a1.output(cout);

cout << "Enter amount of dollars: ";

cin >> d2;

cout << "Enter rate: ";

cin >> r2;

a1.set(d2, r2);

cout << "account1 with new setup:\n";

a1.output(cout);

a1.update( );

cout << "account1 after update:\n";

a1.output(cout);

a2 = a1;

cout << "account2:\n";

a2.output(cout);

return 0;

}

void BankAccount :: set(int d, int c, double r){

if ((d < 0) || (c < 0) || (r < 0)){

cout << "Illegal values for money or interest rate.\n";

return;

}

bal = d + 0.01\*c;

intRate = r;

}

void BankAccount :: set(int d, double r){

if ((d < 0) || (r < 0)){

cout << "Illegal values for money or interest rate.\n";

return;

}

bal = d;

intRate = r;

}

void BankAccount :: update( ){

bal = bal + frac(intRate) \* bal;

}

double BankAccount :: frac(double pv){

return (pv / 100.0);

}

double BankAccount :: getBal( ){

return bal;

}

double BankAccount :: getRate( ){

return intRate;

}

void BankAccount :: output(ostream& outs){

outs.setf(ios :: fixed);

outs.setf(ios :: showpoint);

outs.precision(2);

outs << "Account balance $" << bal << endl;

outs << "Interest rate " << intRate << "%" << endl;

}

**4. Class with constructor and destructor.**

#include <iostream>

using namespace std;

class myclass{

int a;

public:

myclass( );

~myclass( );

void show( );

};

int main( )

{

myclass obj;

obj.show( );

return 0;

}

myclass :: myclass( ){

cout << "In constructor\n";

a = 10;

}

myclass :: ~myclass( ){

cout << "Destructing....\n";

}

void myclass :: show( ){

cout << a << "\n";

}

**5. Time interval of a program using class.**

#include <iostream>

#include <ctime>

using namespace std;

class timer{

clock\_t start;

public:

timer( );

~timer( );

};

int main( )

{

timer obj;

char c;

cout << "Press a key followed by ENTER: ";

cin >> c;

return 0;

}

timer :: timer( ){

start = clock( );

}

timer :: ~timer( ){

clock\_t end;

end = clock( );

cout << "Elapsed time: " << (end - start) /

CLOCKS\_PER\_SEC << "\n";

}

**6. Dynamic memory allocation of a string in class.**

#include <iostream>

#include <cstring>

#include <cstdlib>

#define SIZE 100

using namespace std;

class strtype{

char \*p;

int l;

public:

strtype( );

~strtype( );

void set(char \*ptr);

void show( );

};

int main( )

{

strtype s1, s2;

s1.set("Which is best\n");

s2.set("CSE\n");

s1.show( );

s2.show( );

return 0;

}

strtype :: strtype( ){

p = new char[SIZE];

if(!p){

cout << "Allocation error\n";

exit(1);

}

}

strtype :: ~strtype( ){

cout << "Destructing...\n";

free(p);

}

void strtype :: set(char \*ptr){

if(strlen(p) >= SIZE){

cout << "String too big\n";

}

p = ptr;

l = strlen(p);

}

void strtype :: show( ){

cout << p << "- length: " << l << endl;

}

**7. Overloading of constructor.**

#include <iostream>

using namespace std;

class myclass{

int x;

public:

myclass( ){

x = 0;

}

myclass(int n){

x = n;

}

int getx( ){

return x;

}

};

int main( )

{

myclass o1(10);

myclass o2;

cout << "o1: " << o1.getx( ) << "\n";

cout << "o2: " << o2.getx( ) << "\n";

return 0;

}

**8. Object pointer**

#include <iostream>

using namespace std;

class myclass{

int a;

public:

myclass(int x);

int get( );

};

int main( )

{

myclass ob(120);

myclass \*p;

p = &ob;

cout << "Value using object: " << ob.get( ) << endl;

cout << "Value using pointer: " << p -> get( ) << endl;

}

myclass :: myclass(int x){

a = x;

}

int myclass :: get( ){

return a;

}sss