```cpp
//============================================
// Names: Mayuran Selvarasa, Md Rafi Al Arabi Bhuiyan and Mohammad Yeamin Khan
// Student Number: 019126143,147307193,114964190
// Email:          mselvarasa1@myseneca.ca , mraabhuiyan@myseneca.ca , mykhan10@myseneca.ca
// Section:        DBS211NFF
// Workshop:       Part 1 of Assignment
//============================================

#define _CRT_SECURE_NO_WARNINGS
#include <iostream>
#include <occi.h>
#include <iomanip>
#include <sstream>
#include <string>

#include "Menu.h"

using namespace std;

namespace dbs
{
	int findEmployee(Connection* conn, int employeeNumber, Employee* emp)
	{
		Statement* stmt = nullptr;
		ResultSet* rs = nullptr;

		string query = "SELECT employeenumber, lastname, firstname, email, phone, extension, reportsto, 
jobtitle, city FROM employees JOIN offices ON employees.officeCode = offices.officeCode";

		stmt = conn->createStatement(query);
		rs = stmt->executeQuery();

		if (!rs->next()) {
			// if the result set is empty
			cout << "ResultSet is empty." << endl;
			return 0;
		}
		else
		{
			do {

				int a = rs->getInt(1);

				if (a == employeeNumber)
				{
					cout << "Employee Found!" << endl;
					emp->employeeNumber = employeeNumber;
					strcpy(emp->lastName, rs->getString(2).c_str());
					strcpy(emp->firstName, rs->getString(3).c_str());
					strcpy(emp->email, rs->getString(4).c_str());
					strcpy(emp->phone, rs->getString(5).c_str());
					strcpy(emp->extension, rs->getString(6).c_str());
					strcpy(emp->reportsTo, rs->getString(7).c_str());
					strcpy(emp->jobTitle, rs->getString(8).c_str());
					strcpy(emp->city, rs->getString(9).c_str());

					return 1;
				}

			} while (rs->next());

			cout << "Employee " << employeeNumber << " does not exist." << endl;
			return 0;
		}

		conn->terminateStatement(stmt);
		cout << endl;
	}
void terminate(Environment* env, Connection* conn)
{
	env->terminateConnection(conn);
	Environment::terminateEnvironment(env);
}

void menuTitles()
{
	cout << "*********************HR Menu*********************" << endl;
	cout << "1) Find Employee\n";

	cout << "2) Employees Report\n";

	cout << "3) Add Employee\n";

	cout << "4) Update Employee\n";

	cout << "5) Remove Employee\n";

	cout << "0) Exit\n";
}

int checkValue()
{
	string str = "\0";
	int value = -1;

	do
	{
		cin >> str;
		cout << "Choice is " << str << endl;

		if ((str.compare("1") == 0) || (str.compare("2") == 0)
			|| (str.compare("3") == 0) || (str.compare("4") == 0)
			|| (str.compare("5") == 0) || (str.compare("0") == 0))
		{
			stringstream ss(str);
			ss >> value;
		}

		else
		{
			cout << "Invalid choice, only numbers from 0 to 5 are acceptable, retry: ";
			value = -1;
		}

	} while (value == -1);

	return value;
}

int menu(void)
{
	int value = -1;
	int empNum = 0;

	menuTitles();
	cout << "Enter an option (0-5): ";

	value = checkValue();

	return value;
```

```cpp
            }

            void displayEmployee(Connection* conn, Employee* emp)
            {
                        cout << "\nemployeeNumber = " << emp->employeeNumber << endl;
                        cout << "lastName = " << emp->lastName << endl;
                        cout << "firstName = " << emp->firstName << endl;
                        cout << "email = " << emp->email << endl;
                        cout << "phone = " << emp->phone << endl;
                        cout << "extension = " << emp->extension << endl;
                        cout << "reportsTo = " << emp->reportsTo << endl;
                        cout << "jobTitle = " << emp->jobTitle << endl;
                        cout << "city = " << emp->city << endl;
                        cout << endl;
            }

            void displayAllEmployees(Connection* conn)
            {
                        // Defining Objects
                        Statement* stmt = nullptr;
                        ResultSet* rs = nullptr;

                        string query = "SELECT  e.employeenumber, e.firstname || ' ' || e.lastname AS empName, e.email, phone,
e.extension, em.firstname || ' ' || em.lastname AS manName FROM employees e JOIN offices o ON e.officecode = o.officecode
LEFT JOIN employees em ON e.reportsTo = em.employeeNumber ORDER BY employeenumber";

                        stmt = conn->createStatement(query);
                        rs = stmt->executeQuery();

                        const char separator = '  ';
                        const int width1 = 15;
                        const int width2 = 20;
                        const int width3 = 35;

                        cout << "Displaying Employee Report" << endl;

                        cout << left << setw(width1) << setfill(separator) << "\nID";
                        cout << left << setw(width2) << setfill(separator) << " Employee Name";
                        cout << left << setw(width3) << setfill(separator) << " Email";
                        cout << left << setw(width2) << setfill(separator) << " Phone";
                        cout << left << setw(width1) << setfill(separator) << " Extension";
                        cout << left << setw(width1) << " Manager Name" << endl;
                        char oldFill = cout.fill('-');
                        cout.width(115);
                        cout << "";
                        cout.fill('-');
                        cout << endl;

                        if (!rs->next()) {
                                    // if the result set is empty
                                    cout << "There is no employees� information to be displayed." << endl;
                        }
                        else
                        {
                                    do
                                    {
                                                cout << left << setw(width1) << setfill(separator) << rs->getInt(1);
                                                cout << left << setw(width2) << setfill(separator) << rs->getString(2);
                                                cout << left << setw(width3) << setfill(separator) << rs->getString(3);
                                                cout << left << setw(width2) << setfill(separator) << rs->getString(4);
                                                cout << left << setw(width1) << setfill(separator) << rs->getString(5);
                                                cout << left << setw(width1) << rs->getString(6) << endl;
                                                cout << endl;
                        } while (rs->next());
                }
                conn->terminateStatement(stmt);
        }

        int isDigit()
        {
                char str[] = "\0";
                int x = -1;

                do
                {
                        x = 1;
                        cin >> str;

                        for (int i = 0; i < strlen(str); i++)
                        {
                                // check for alphabets
                                if (isalpha(str[i]) != 0)
                                {
                                        cout << "Only an integer value is acceptable, retry : ";
                                        x = -1;
                                        break;
                                }
                        }
                } while (x == -1);

                int y = atoi(str);
                return y;
        }
}
```