**12 Створення програмних модулів мовою VBA**

##### 12.1 Основи програмування мовою VBA у середовищі СКБД Access

Код VBA слід використовувати замість макросів для виконання зазначених нижче задач:

* **Використання вбудованих або створення власних функцій**   Access включає багато вбудованих функцій, таких як функція **Ipmt**, яка розраховує суму процентних платежів. Вбудовані функції можна використовувати для розрахунків без створення складних виразів. За допомогою коду VBA можна також створювати власні функції для виконання розрахунків, що перевищують можливості виразів, або для заміни складних виразів. Крім того, можна використовувати створені функції у виразах для застосування загальних операцій до декількох об'єктів.
* **Створення об'єктів або маніпулювання ними.**    У більшості випадків найбільше зручно створювати й змінювати об'єкт у режимі конструктора об'єкта. Однак у деяких ситуаціях може знадобитися виконати ті або інші дії з визначенням об'єкта за допомогою коду. VBA дозволяє маніпулювати всіма об'єктами в базі даних, а також базою даних у цілому.
* **Виконання дій на рівні системи.**    У макросі можна виконати дію **ЗапускПриложения** для запуску іншої програми (наприклад, Microsoft Excel) з Access, але макрос не можна використовувати для виконання операцій поза Access. За допомогою VBA можна перевірити наявність файлу на комп'ютері, використовувати автоматизацію або динамічний обмін даними (DDE) для взаємодії з іншими програмами Windows, наприклад Excel, або викликати функцію з бібліотеки динамічного компонування (DLL) Windows.
* **Керування окремими записами.**    VBA можна використовувати для послідовного проходу по набору записів і виконання операції для кожного запису, тоді як макроси працюють із наборами записів.

##### Основні компоненти програми на VBA – це процедури та функції. Синтаксис процедури наступний:

SUB <NameProc>(<argument1>, <argument2>, …)

<оператори VBA>

END SUB

А синтаксис функції наступний:

FUNCTION <NameFunc>(<argument1>, <argument2>, …)

<оператори VBA>

END FUNCTION

Різниця між SUB i FUNCTION:

* у тілі функції хоча б один раз повинен з’явитися оператор присвоєння результату імені функції;
* ім’я функції з фактичними параметрами можна використовувати як звичайну змінну у операторах VBA.

Коди VBA в Access збираються у модулі. Існують два типи модулів **–** *стандартні* та *модулі класу.*

*Стандартні модулі* – це загальні модулі, що не пов’язані з об’єктами бази даних: формами або звітами. Перелік стандартних модулів можна побачити, вибираючи об’єкт Модулі (Modules) у вікні бази даних.

Нижче наведено приклад стандартного модуля Sub matr()з виведенням даних у таблицю result.Нагадаємо, що за граматикою VBA коментари-рядки починаються з символу „'„.

Sub matr()

' Мета програми – з двох одновимірних (4 та 2 елементи) масивів знайти

' М3=М1 intersection М2, результат вивести у масив RES, опис якого наведено нижче

Dim res(1) As String

' Зрозуміло, що результат буде мати =< 2 елементів

' Нижче наводиться оператори опису допоміжних змінних:

' N1– лічильник рядків М1

' N2 – лічильник рядків М2

' N3 – лічильник рядків RES

' NN – для виведення значення результату

Dim N1 As Integer

Dim N2 As Integer

Dim N3 As Integer

Dim NN As Integer

' Формування масиву M1

Dim gr1(3) As String

gr1(0) = 12

gr1(1) = 22

gr1(2) = 23

gr1(3) = 33

' Формування масиву M2

Dim gr2(1) As String

gr2(0) = 22

gr2(1) = 33

' Створюємо таблицю для результату з полем «результат»

sql1 = "create table result (результат integer )"

DoCmd.RunSQL sql1

N1 = 0

N2 = 0

N3 = 0

' Реалізація алгоритму

Do While N1 < 4

Do While N2 < 2

If gr1(N1) = gr2(N2) Then

res(N3) = gr1(N1)

NN = res(N3)

'видаємо результат на екран

MsgBox "Елемент вихідного масиву = " & CStr(res(N3))

'записуємо результат у таблицю

'УВАГА!!! Змінна в SQL записується “ & NN & ” – для чисельного значення, а

' для текстового пишется ‘ “ & NN & “ ‘!!!

DoCmd.RunSQL ("INSERT INTO result VALUES( " & NN & " )")

N3 = N3 + 1

End If

N2 = N2 + 1

Loop

N2 = 0

N1 = N1 + 1

Loop

MsgBox "Кількість елементів = " & CStr(N3)

' Переглядаємо результат!!!

DoCmd.OpenTable "result", acViewNormal, acReadOnly

MsgBox "Результат роботи в таблиці. Таблиця закривається"

DoCmd.Close acTable, "result"

' Видаляємо таблицю

DoCmd.RunSQL ("drop table result")

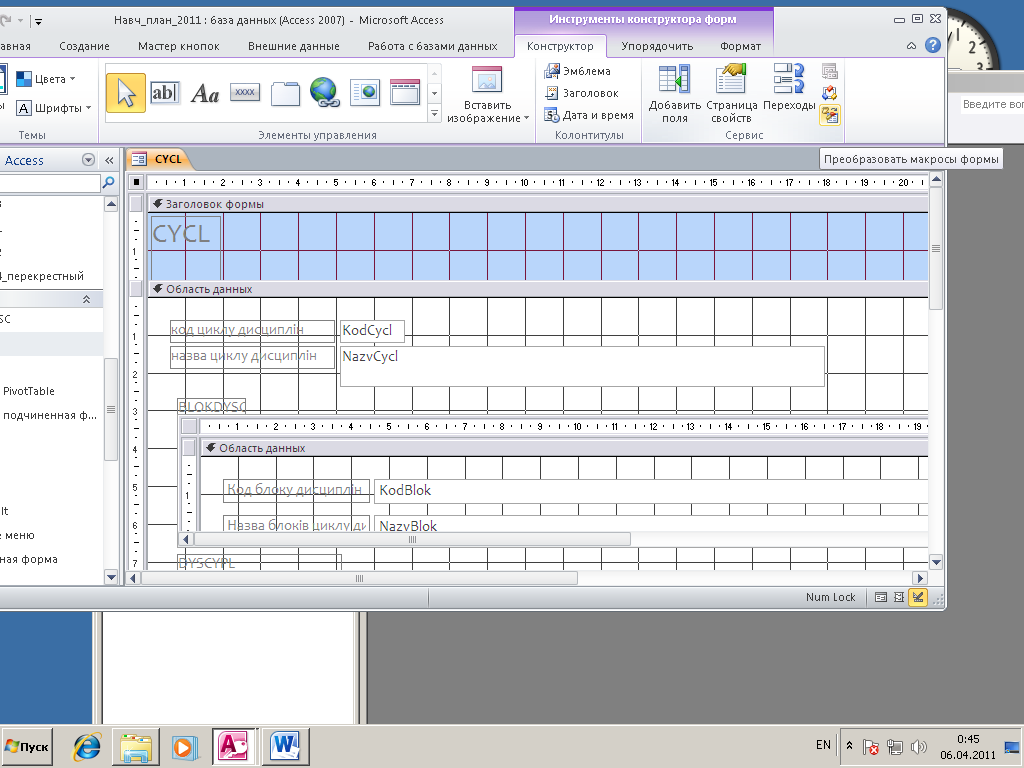
End Sub

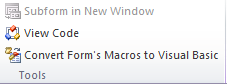
Тепер розглянемо другий тип модулів – м*одулі класу*.Вони, крім коду процедур, мають ще опис об’єкту , тому це модулі форм та звітів. Вони пов’язані з конкретними формами або звітами та вміщують процедури обробки подій.

### Перетворення макросів, які вбудовані у форму або звіт в код VBA

Будь-які макроси, які згадуються у формі або звіті (або в будь-якому їхньому елементі керування) або вбудовані в них можна перетворити у код VBA. Access додає код VBA у модуль класу форми або звіту. Модуль класу стає частиною форми або звіту й переміщається з ними при їхньому копіюванні або переміщенні.

1. В області навігації клацніть правою кнопкою миші форму або звіт і виберіть пункт **Конструктор**.
2. На вкладці **Конструктор** у групі **Сервіс** виберіть команду **Перетворення макросів форми в Visual Basic** або **Перетворення макросів звіту в Visual Basic**.





1. У діалоговім вікні **Перетворення макросів форми** або **Перетворення макросів звіту** вкажіть, чи треба додати до створюваних функцій код обробки помилок. Крім того, при наявності в макросах коментарів укажіть, чи включати їх у функції. Для продовження натисніть кнопку **Перетворити**.
2. Якщо модуль класу для форми або звіту відсутній, Access створить його й додасть у нього процедуру для кожного макросу, пов'язаного з формою або звітом. Крім того, Access змінить властивості подій форми або звіту так, щоб вони запускали нові процедури VBA замість макросів.

**Приклад модуля класу:**

Алгоритм роботи

1.Взяти 2 зв'язані таблиці (CYCL, DYSCYPL), побудувати вкладену форму

2.Зробити запит по DYSCYPL - 2 поля KodCycl, KodDysc з кількістью

дисциплін для KodCycl=1 , запит Make table Res2.

3.Виконати запит, щоб створилась таблиця Res2, а SQL-код запиту використати для створення запиту у модулі.

4.У вкладену форму вставити **кнопку** (без застосування Майстра кнопок),

виділити об'єкт **кнопку**, правою кнопкою миші

відкрити контекстне меню, через Побудувач Подій (Build Events)

відкрити Code і написати програму, приклад якої наведен нижче

УВАГА! Назва кнопки **Кнопка6** у вас може бути іншою**!!!**

**Private Sub Кнопка6\_Click()**

Dim N1 As String

Dim N2 As Variant

Dim N3 As Integer

' Запам'ятати значення коду цикла для підрахунку у змінній N1, виправити ім'я форми '(Cycl), якщо воно інше!

N1 = [Forms]![Cycl]![ KodCycl]

' зробити підрахунок по конкретному значенню KodCycl

'УВАГА! Символьна параметрична змінна вказується як '" & N1 & "'

' оператори можна писати у 2 рядки через "\_"

' Змінна KodCycl у нас цифрова, тому в цьому запиті вказуємо " & N1 & "

sql1 = "SELECT DISTINCTROW Count(DYSCYPL.KodDysc) AS [Всього дисциплін], DYSCYPL.KodCycl " \_

&"INTO res2 FROM DYSCYPL GROUP BY DYSCYPL.KodCycl HAVING (((DYSCYPL.KodCycl) = " & N1 & "))"

DoCmd.RunSQL sql1

' Функція DLookup вибирає значення поля з таблиці

N2 = DLookup("[Всього дисциплін]", "res2")

If Not IsNull(N2) Then

MsgBox "Всього дисциплін =" & N2, , "РОЗРАХУНОК"

Else

MsgBox "Немає даних", 0, "ПОМИЛКА!"

End If

'Видалити тимчасову таблицю

DoCmd.DeleteObject acTable, "res2"

End Sub

**Приклади функції DLookup**

Sub proba1()

'Приклад використання функції DLookup з умовою

Dim varX As Variant

DoCmd.OpenForm "res2", acNormal, , , acFormAdd, acHidden

varX = DLookup("[Всього дисциплін]", "res3", \_

"[KodCycl]= forms!res2![KodCycl]")

MsgBox "Результат =" & varX

DoCmd.Close acForm, "res2"

End Sub

Sub proba2()

'Приклад використання функції DLookup з умовою тиnу Number

Dim intSearch As Integer

Dim varX As Variant

intSearch = 1

varX = DLookup("[Всього дисциплін]", "res3", \_

"[KodCycl]= " & intSearch)

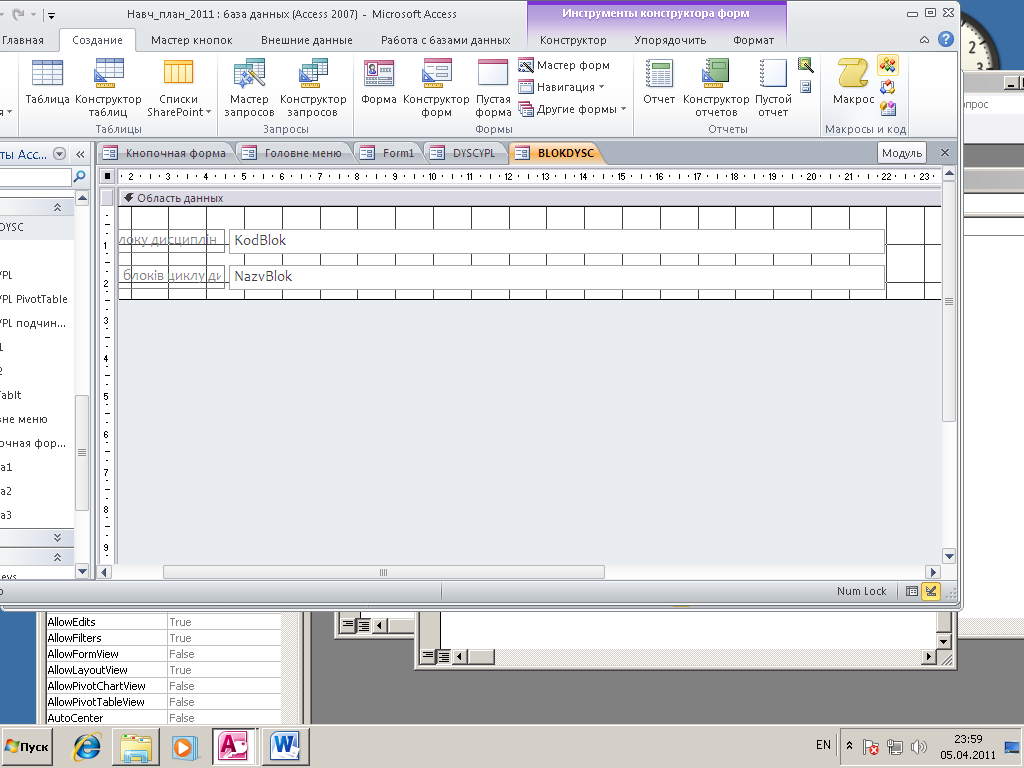
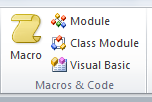
MsgBox "Результат =" & varX

End Sub

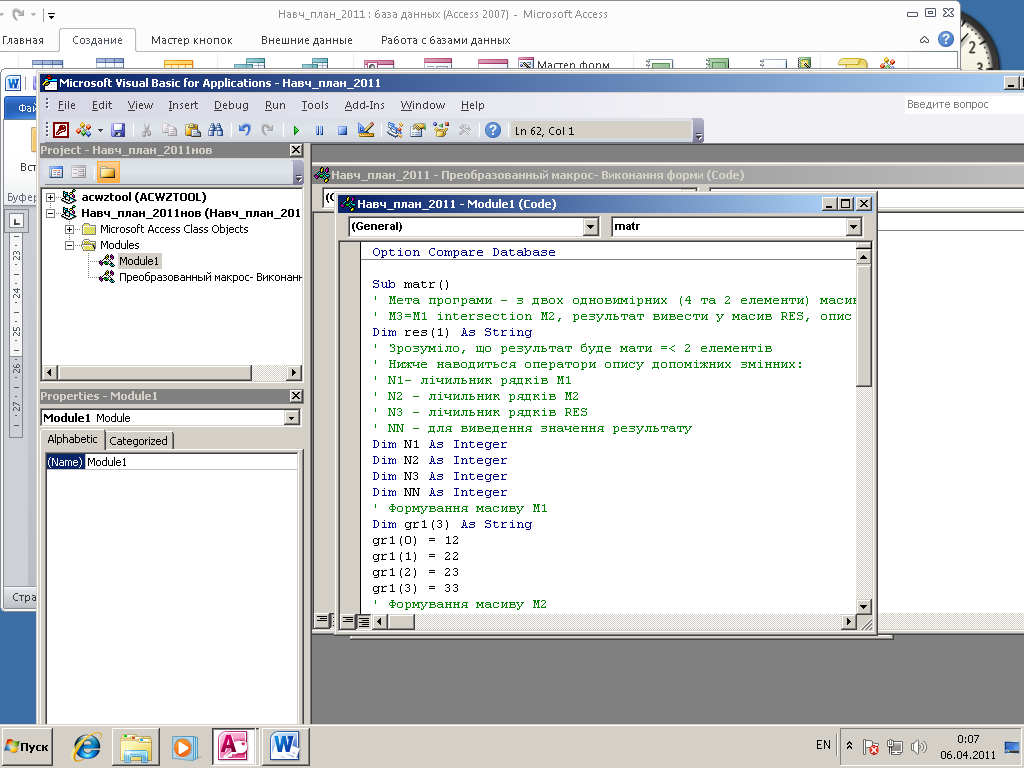
##### 12.2 Лабораторна робота №12. Завдання

##### Створення стандартного модуля

1. На вкладці **Создание/Create** вибрати команду Модуль

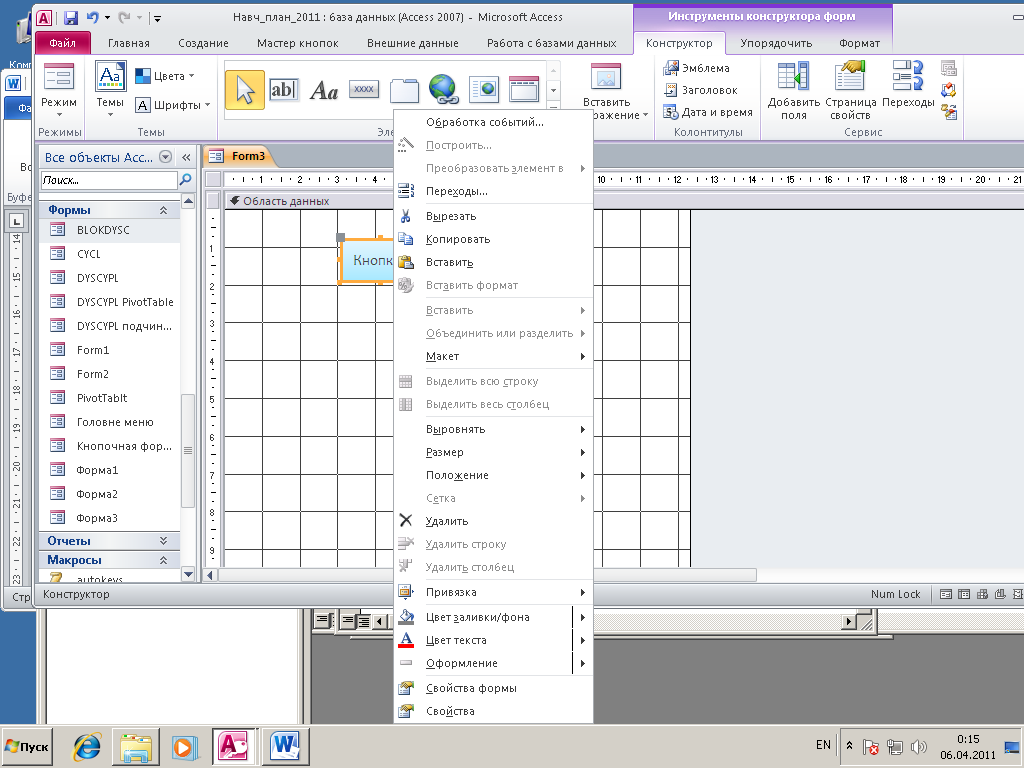
 

1. Створюється новий модуль у вікні VBA, в який скопіювати програму **matr**



Виконати DEBUG – Compile

1. Виконати процедуру командою **Run**
2. Створити кнопкову форму, при появі **Wizard** кнопки вибрати **ОТМЕНА/Cancel.** Вибрати кнопку, правою клавішею миші відкрити контекстне меню, вибрати **Обработка событий/Build Events**



У вікні, що з’явиться, вибрати **Програми/Code**, в процедурі вказати назву **matr**

**Private Sub Кнопка0\_Click()**

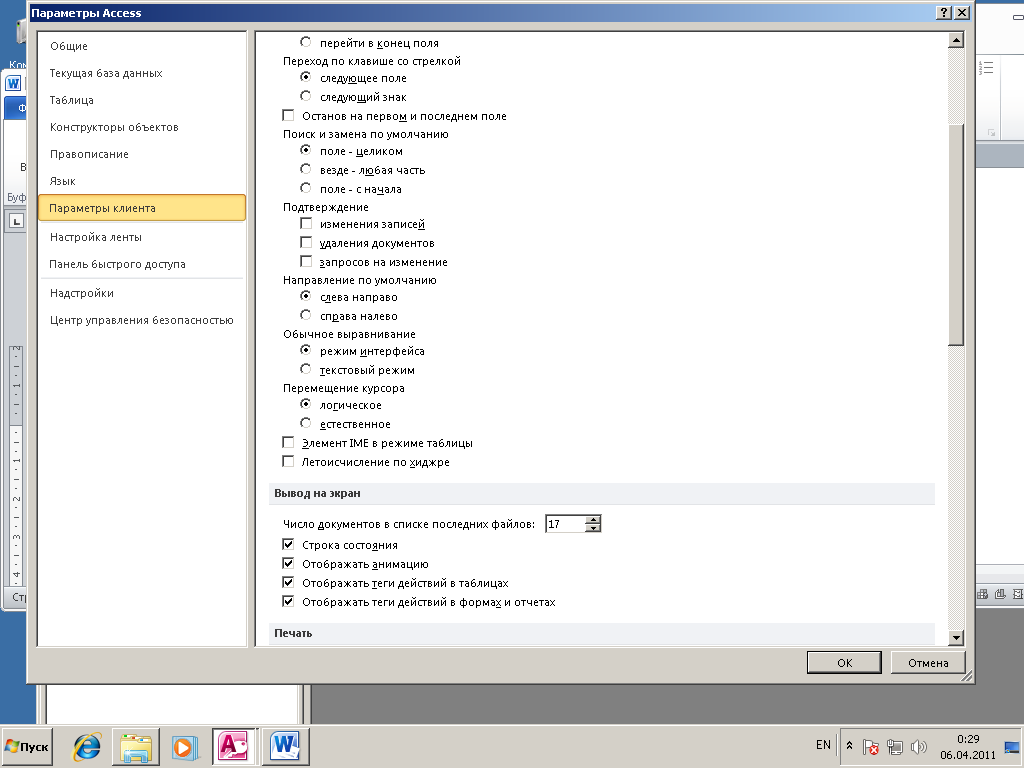
**matr**

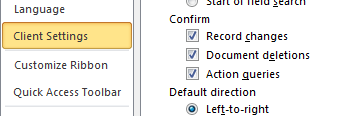
**End Sub**

Виконати DEBUG – Compile

Зберегти форму, виконати.

1. Для припинення виведення на екран підтвердження видалення або зміни даних зминіть параметри на вкладці **Файл/File** команда **Параметри/Options** підкоманда **Параметри клієнта/Client Setting –** зняти прапорці у групі підтвердження





1. Повторити виконання модуля. Порівняти результати.

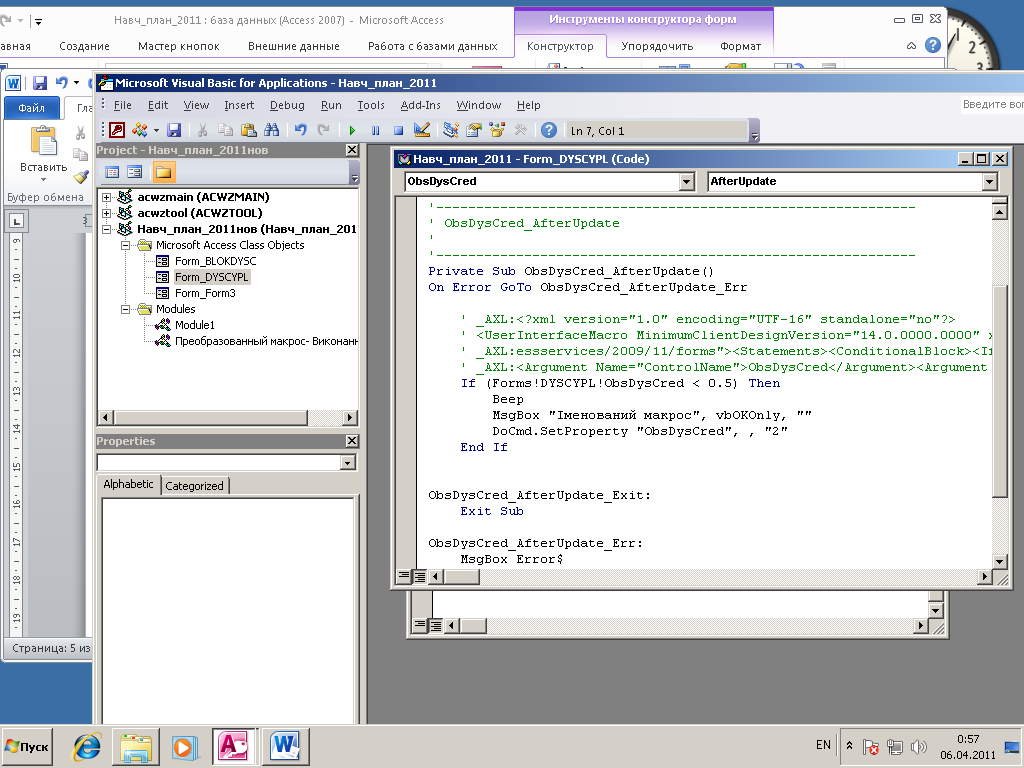
##### Налагодження модулів

1. Відкрити модуль з процедурою **Matr.** Включити режим покрокового налагодження **Debug-Step Into,** включити режим  **Watch** для змінної **NN1,** натискуючи на **F8** виконати налагодження.

***Модулі класу***

1. Відкрити форму для DYSCYPL, створену в п.1 Лабораторної роботи 11 в режимі Конструктор, перетворити макрос в цій формі у модуль класу.

Отримаємо результат



Виконати DEBUG – Compile

1. Виконати форму, переглянути значення в таблиці, відкоригувати значення поля ObsDysCred на 0,3 (повинно з’явитися повідомлення і відкоригуватися значення ObsDysCred на 2.
2. Створити форму і модуль класу, як наведено в прикладі модуля класу.
3. Кінець