МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет “Львівська політехніка”

Засоби створення додатків.
СЕРЕДОВИЩЕ ПРОГРАМУВАННЯ MS Visual C++ 2010

ІНСТРУКЦІЯ

до лабораторної роботи № 1 з курсу
“Основи програмування”
для базового напрямку “Програмна інженерія”

Затверджено
На засіданні кафедри
програмного забезпечення
Протокол № від

ЛЬВІВ – 2011

1. МЕТА РОБОТИ

Мета роботи – ознайомитися з основами роботи в інтегрованому середовищі Microsoft Visual C++ 2010 Express Edition, а саме з його можливостями для введення, відлагодження та виконання програм на мові C/C++.

2. ТЕОРЕТИЧНІ ВІДОМОСТІ

Мова С є однією з найбільш поширених серед сучасних мов програмування. Завдяки високій ефективності, потужності функціональних можливостей та мобільності, застосування мови С має досить широкі області. Основна філософія мови С грунтується на тому що програміст знає, що робить і явно вказує ці наміри. Однак, для того, щоб написати та виконати програму на мові С++ потрібно використати певне середовище програмування (англомовний термін IDE – Integrated Development Environment), таке як, наприклад, Microsoft Visual C++ 2010 Express Edition (надалі скорочено MS VC++). До складу такого середовища входять засоби для набору та редагування тексту програми, її компіляції, відлагодження та запуску на виконання, які значно спрощують роботу програміста. Саме тому, перед безпосереднім вивченням азів програмування тією чи іншою мовою програмування, доцільно оволодіти основними навиками роботи в такому середовищі, яке в подальшому буде використовуватися як інструмент роботи програміста.

Запуск MS VC++ відбувається стандартним для будь-якого Windows-застосування способом (через стартове меню або через відповідну піктограмку на робочому столі). Після успішного завантаження на екрані монітора з’явиться головне вікно програми MS VC++, яке зображено на рис.1.

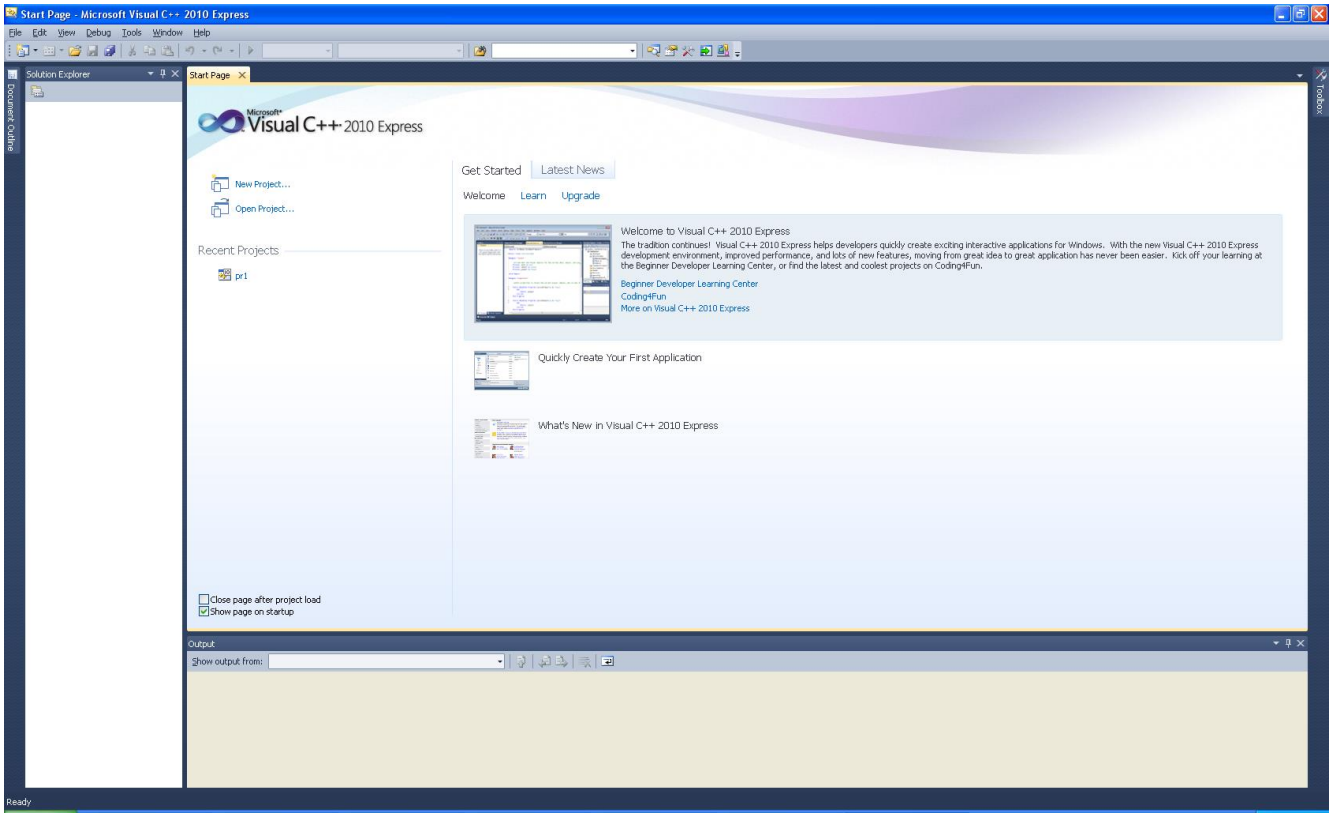


Рис. 1. Головне вікно IDE Microsoft Visual C++ 2010 Express Edition.

У верхній частині вікна розташована стрічка з *командами головного меню* IDE MS VC++ (команди **File**, **Edit**, **View** і т.д.) – стрічка горизонтального меню. При виклику цих команд відкриваються так звані “випадаючі меню” – це вертикальні меню, які складаються з набору команд, розташованих на екрані зверху вниз. Приклад такого меню зображено на рис.2.

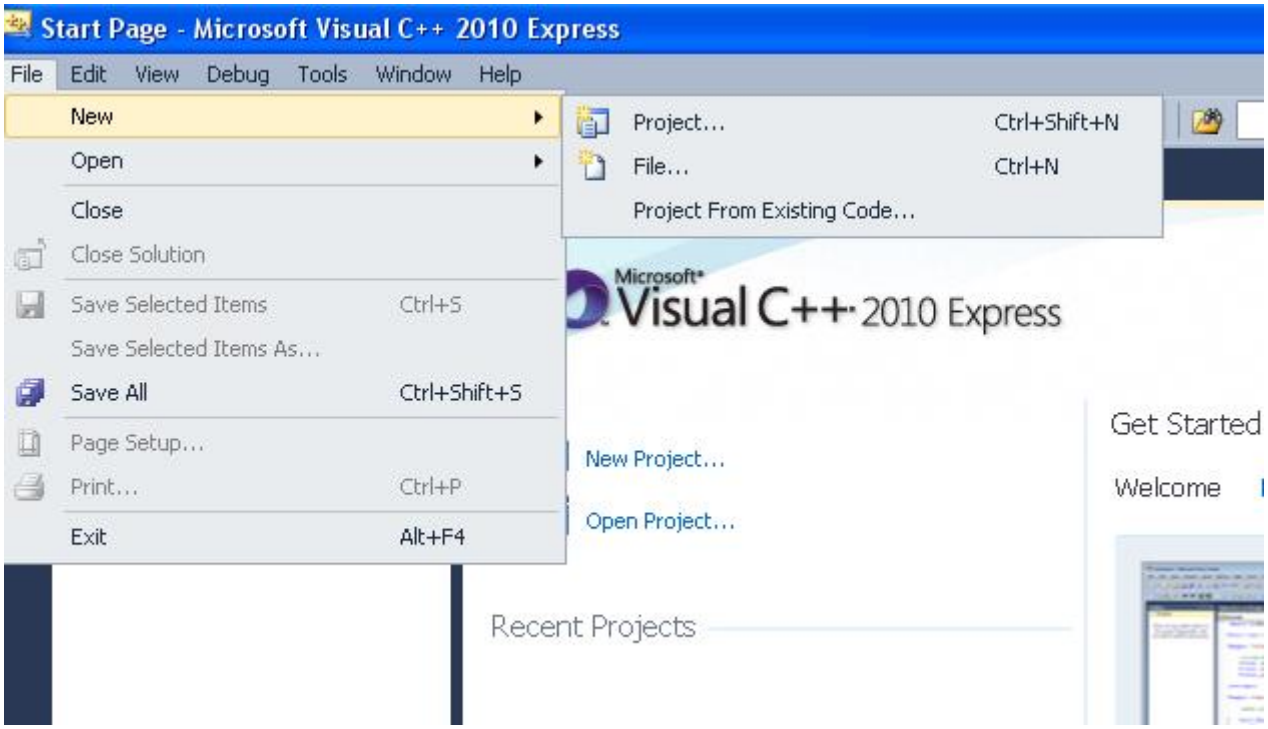


Рис.2. Горизонтальне головне меню і вертикальне випадаюче меню

У другому рядку головного вікна, розташованому під стрічкою головного меню, знаходиться панель інструментів, яка містить кнопки *швидкого виклику* деяких найбільш часто вживаних команд меню. Всі ці кнопки мають *спливаючі підказки* (tooltips) про призначення кнопки, які автоматично з'являються при наведені на них курсора миші. Поряд з такими кнопками можуть бути додаткові кнопки, у вигляді маленького чорного трикутника з вершиною донизу, які призначені для розкриття списку можливих значень основної кнопки. Оскільки всі кнопки не поміщаються у відведене для них місце в головному вікні, то вони згорнуті в невеличкі полоски з кнопками для їх розгортання, точно так само, як у добре відомому текстовому редакторі MS Word (рис. 3).

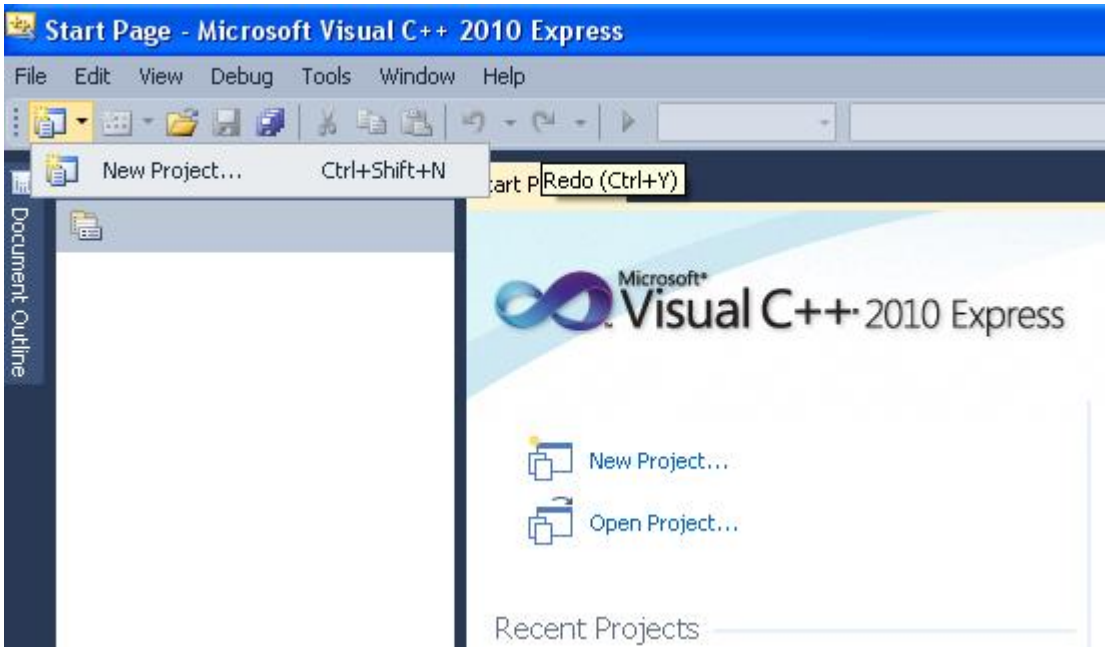


Рис.3. Кнопки швидкого доступу стандартної панелі інструментів

Основна частина головного вікна IDE MS VC++ - робоча область, вигляд якої залежить від типу прикладної програми, яку розробляє програміст. Тим не менше, робоча область

містить декілька постійних складових частин, до яких у першу чергу відносяться вікно **Solution Explorer** (провідник рішення) та вікно стартової сторінки (**Start Page**). Стартова сторінка складається з декількох частин, найбільш важливі з них знаходяться в її лівій частині. Тут є область яка містить команди створення нового проекту (**New Project**), відкриття існуючого проекту (**Open Project**) та область списку останніх проектів, з яким працював розробник (**Recent Projects**) (рис.4).

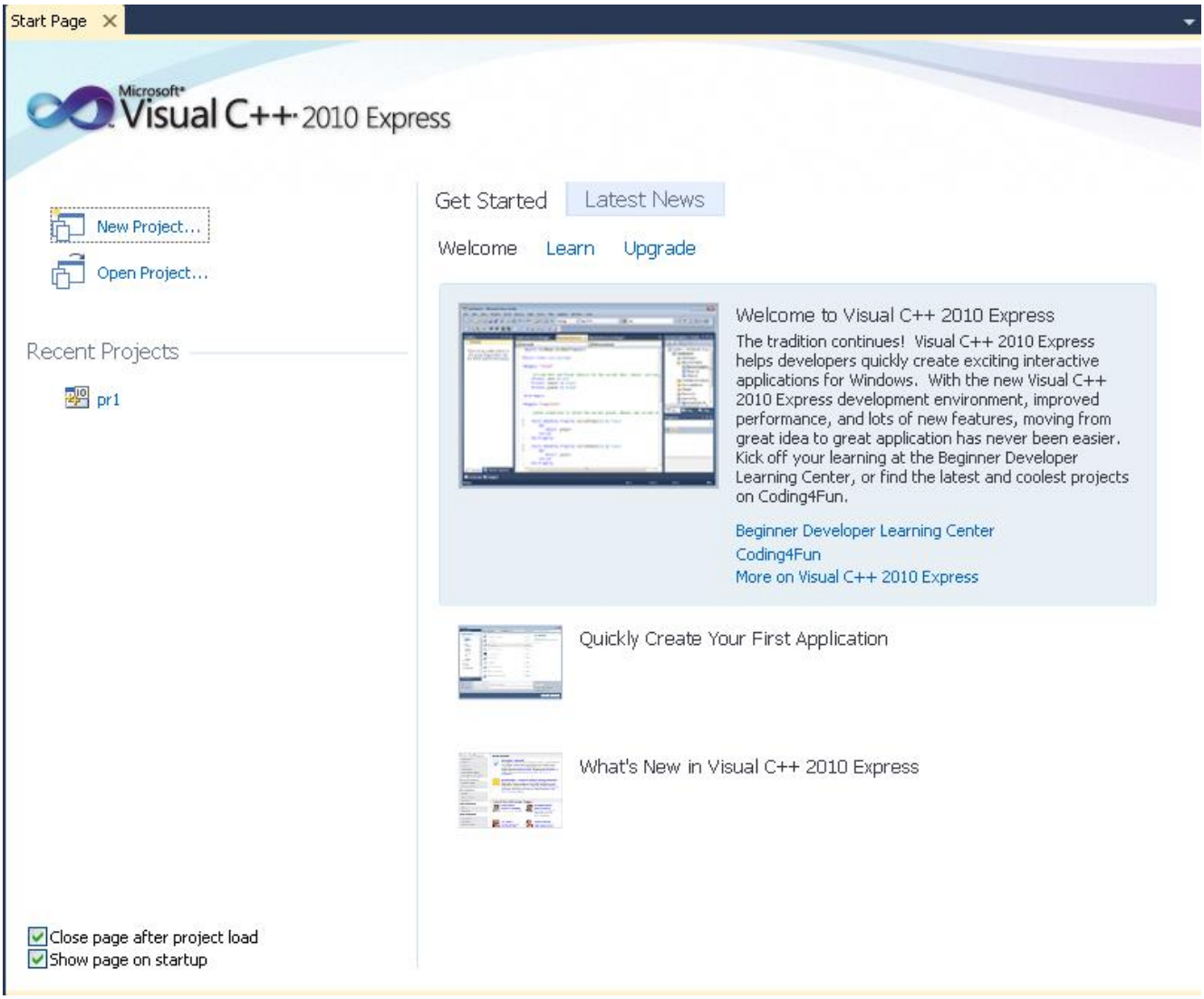


Рис.4. Стартова сторінка

Слід також звернути увагу на дві опції типу прапорець (галочка) у нижній лівій частині стартової сторінки: **Close page after project load** та **Show page on startup**. Включення першої з них (має бути відзначена галочкою) означає, що стартове вікно автоматично закриється після завантаження проекту, а включення другої – що стартова сторінка буде завжди відображатися першою після запуску самого середовища MS VC++ (при бажання її можна відключити знявши галочку).

Для полегшення розуміння структури та призначення вікна **Solution Explorer** доцільно створити першу працюючу програму на мові C++. Самі програми на C++ в середовищі MS VC++ прийнято називати *застосуваннями* (очевидно мається на увазі застосування до самого IDE), від англійського терміну ***application***. Оформлення розробки такого застосування здійснюється автоматично самим IDE MS VC++ у вигляді спеціальної конструкції, яка називається *рішенням* (***solution***). Саме рішення є лише контейнером для так званих *проектів* (***projects***) в рамках яких розробляється саме застосування або його деяка складова частина (у випадку великого і складного застосування). Сам проект виглядає для користувача як

сукупність певних файлів, кожний з яких має своє конкретне призначення і свій вклад в саму програму (застосування). Рішення може складатися з множини проектів, але одне застосування має розроблятися в рамках лише одного рішення. Іншими словами, рішення – це просто група взаємозв'язаних проектів, кожний з яких містить елементи, які забезпечують існування та функціонування застосування як одного цілого. Такий підхід до організації процесу розробки застосувань дозволяє працювати з групою залежних проектів як з одним цілим, що суттєво прискорює розробку прикладних програм, особливо в тих випадках, коли розробка ведеться командою програмістів. Структура рішення і, відповідно, проектів, які до нього входять відображається у вікні **Solution Explorer** робочої області головного вікна MS VC++.

Для створення першої програми на мові C++ достатньо використати спеціальний тип застосування, яке називається *консольним застовуванням* (***Console Application***), на основі існуючих в середовищі MS VC++ заготовок, які прийнято називати *шаблонами*. Консольне (тобто базове) застосування – це програма без графічного інтерфейсу користувача, до якого ми всі звикли в операційних системах сімейства Windows. Консольні застосування взаємодіють з користувачем через спеціальне вікно командного рядка (консольне вікно), в яке виводяться результати виконання програми і через яке користувач з команного рядка може ввести свої вхідні дані для програми. Це вікно автоматично відкривається самим середовищем після запуску застосування, і також автоматично закривається після завершення роботи програми.

Перейдемо власне до створення консольного застосування в середовищі MS VC++. Для цього потрібно виконати такі дії:

1. Відкрити діалогове вікно для створення нового проекту. Це можна зробити декількома способами: або вибрати команду головного меню **File->New->Project,** або на стартовій сторінці виконати команду **New Project**, яка має вигляд лінку (див. рис.4), або за допомогою комбінації "гарячих клавіш" **Ctrl+Shift+N**. У будь-якому випадку відкриється діалогове вікно, зображене на рис.5.
2. У цьому вікні послідовно виконати дії в зазначеному порядку.

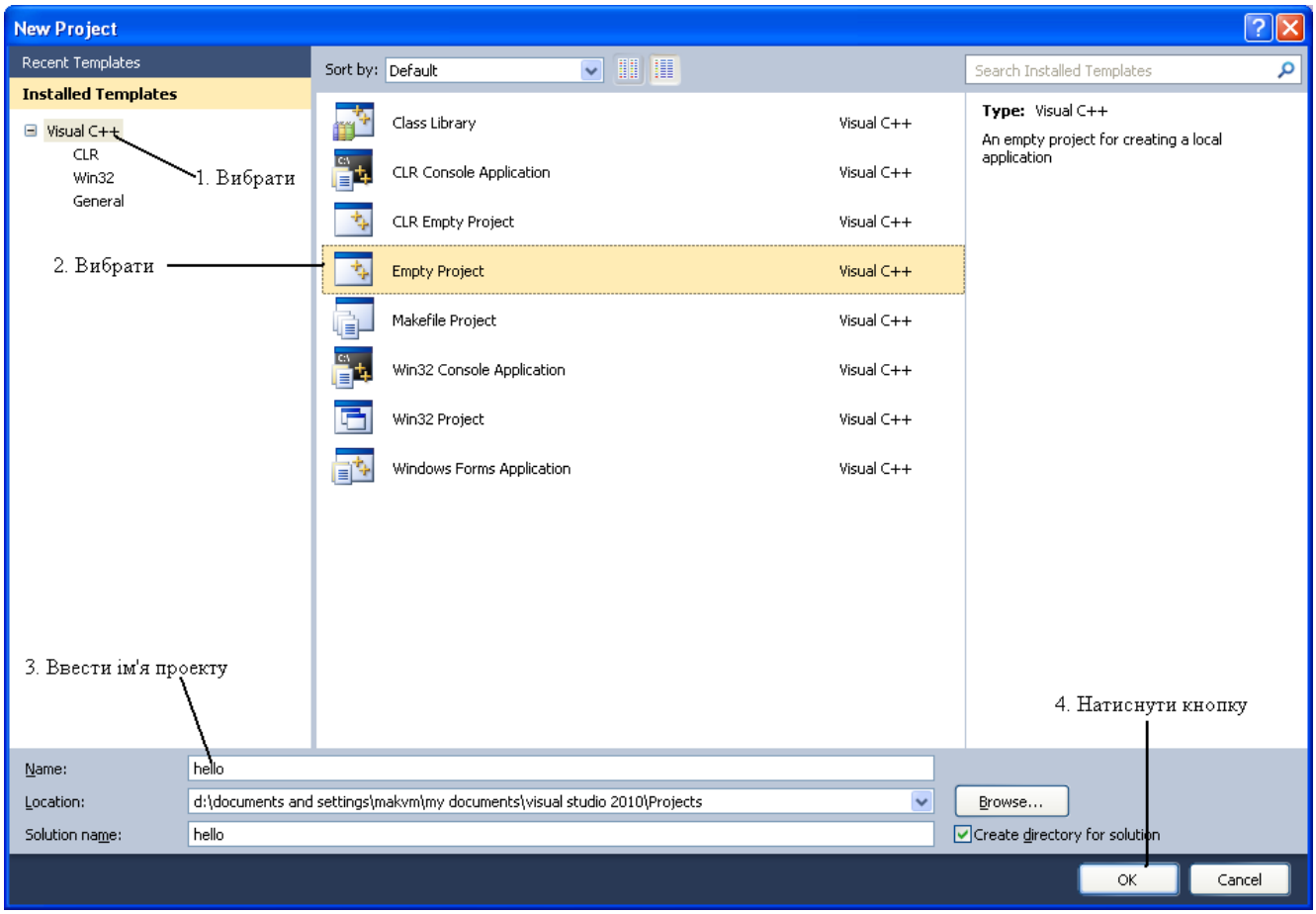


Рис.5. Створення консольного застосування на основі шаблону порожнього проекту

Після виконання цих дій стартова сторінка закриється (оскільки ми залишили увімкненою опцію **Close page after project load),** а у вікні **Solution Explorer** робочої області буде відображено структуру рішення та, відповідно, проекту, так як це показано на рис. 6. Зауважимо, що назва рішення співпадає з назвою проекту, яку ми задали при його створенні, а сам проект не містить жодних файлів (оскільки ми вибрали шаблон **Empty Project**).

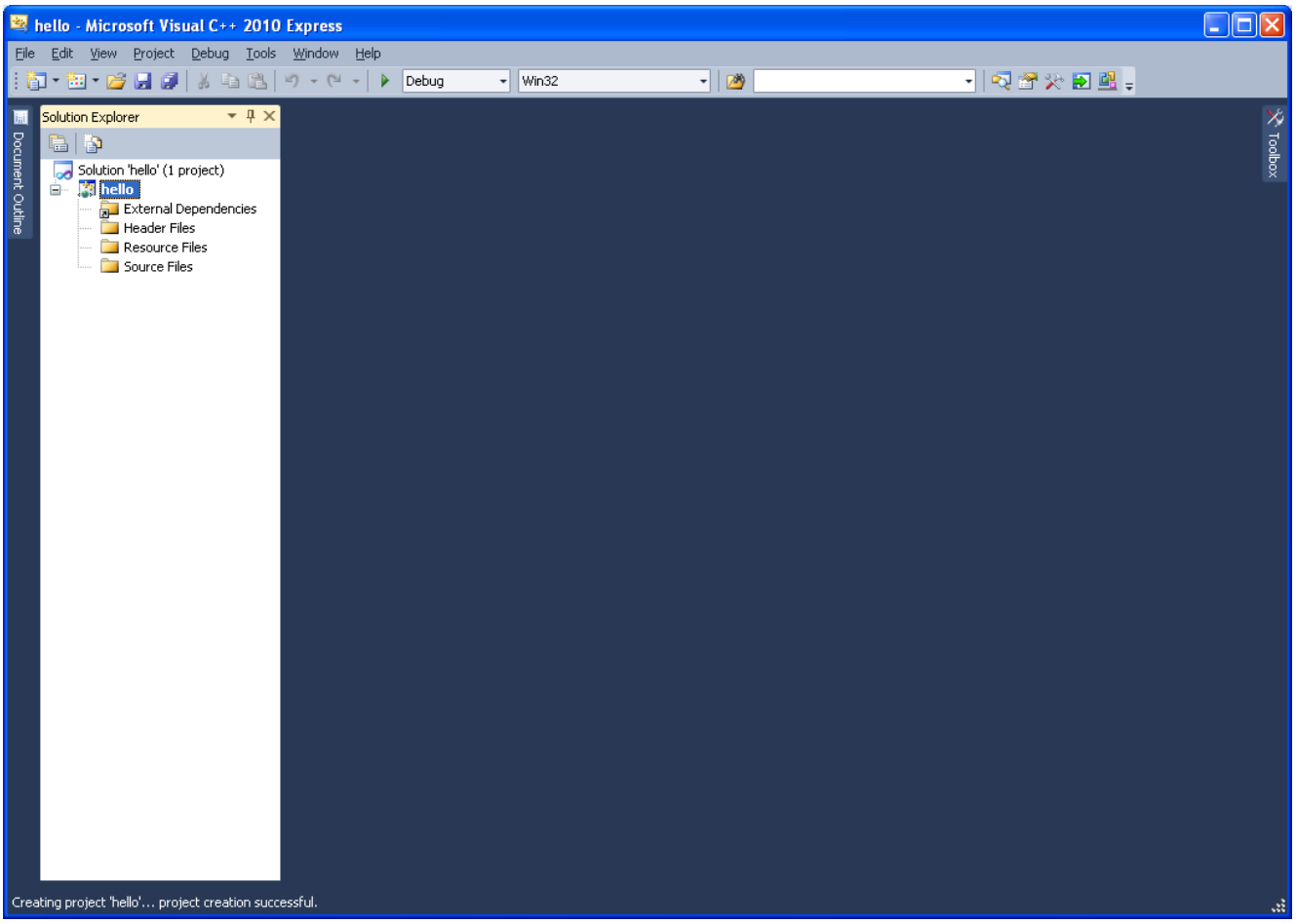


Рис.6. Відображення структури рішення і проекту у вікні **Solution Explorer**

На даний момент ми створили лише каркас нашого першого застосування (іншим словами можна сказати, що ми, поки що, вирішили тільки організаційні питання процесу створення застосування). Тепер ми готові наповнити цей каркас вихідним текстом програми на мові C++ (нарешті приступити до конкретної роботи з програмування). Ці вихідні тексти програми зберігаються у файлах з розширенням .cpp і в структурі проекту зберігаються в частині, яка називається **Source Files**. Оскільки наш проект є порожнім, ця його частина також не містить жодних файлів з вихідними текстами програми. Тому спочатку потрібно додати в проект новий файл, в якому буде знаходитися текст нашої програми. Для цього потрібно послідовно виконати такі дії:

- правою кнопкою миші клікнути на пункті **Source Files** у вікні **Solution Explorer;**
- в контекстно-залежному меню, яке з'явиться на екрані, вибрати команду **Add->New Item** (контекстно-залежне меню – це вертикальне меню, склад команд якого залежить від того, в якому місці воно було викликано; виклик цього меню завжди здійснюється кліком правої кнопки миші);
- у діалоговому вікні **Add New Item**, яке з'явиться на екрані послідовно виконати дії в зазначеному порядку (див. рис.7).

Після виконання цих дій робоча область головного вікна середовища має мати вигляд аналогічний до зображеного на рис.8. Зауважимо, що у вікні **Solution Explorer** в розділі **Source Files** з'явився елемент, який відповідає щойно створеному файлу програмного коду

(**program.cpp**), а в основній частині робочої області автоматично відкрилося вікно редактора програмного коду з вкладкою (з самого верху, містить ім'я файлу) для цього файлу (яке наразі є порожнім) і курсором (у вигляді блимаючої вертикальної риски) у першому рядку вікна.

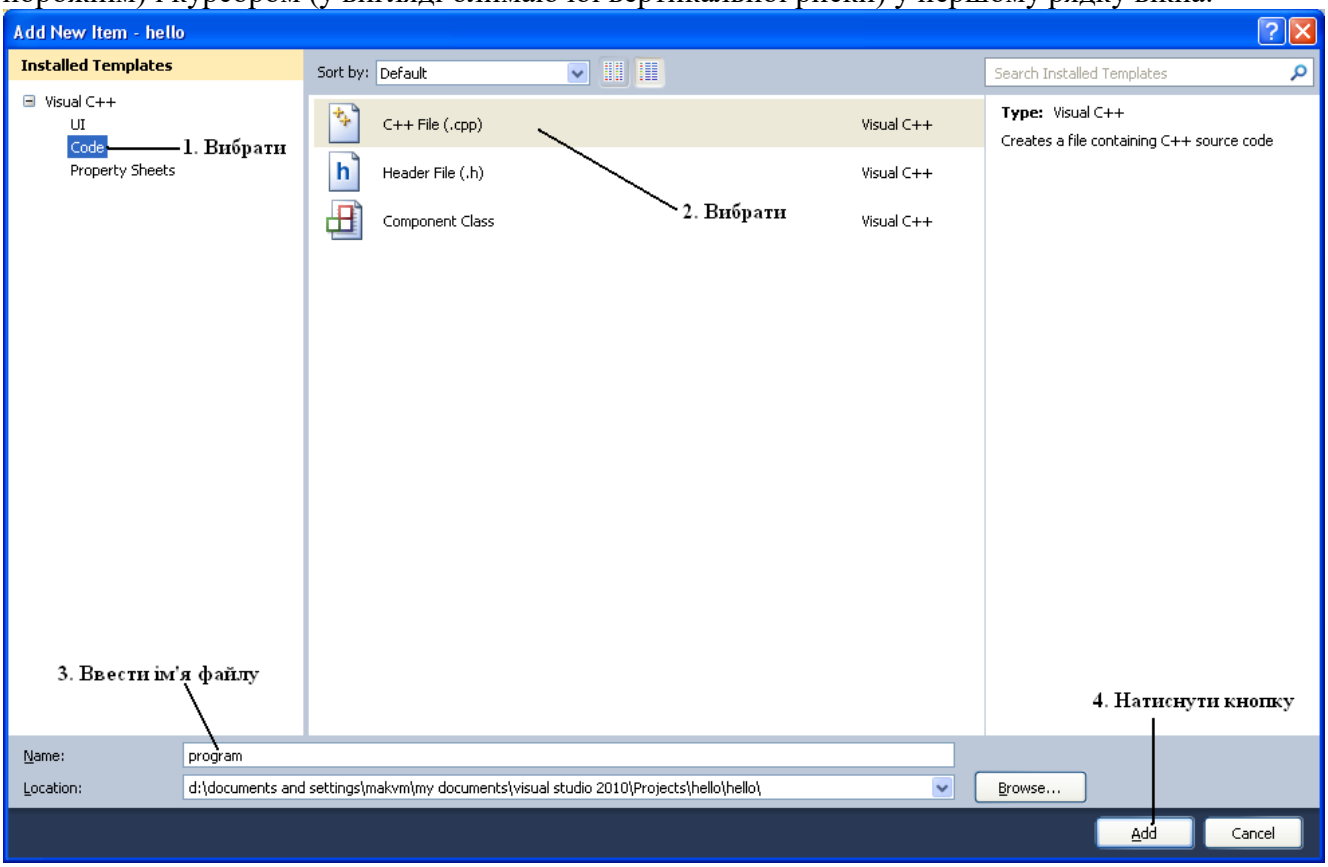


Рис.7. Додавання файлу програмного коду в проект

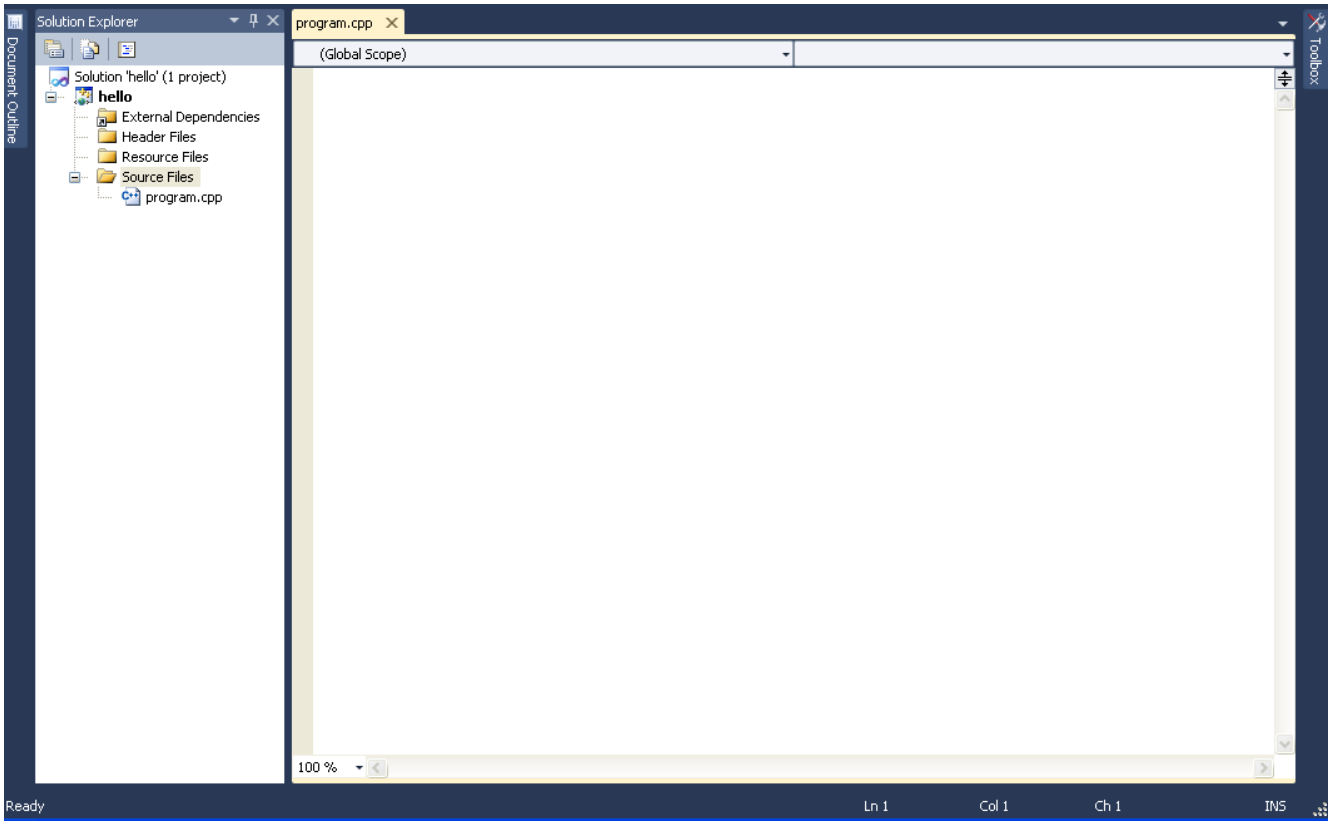


Рис.8. Середовище MS VC++ готове до створення програмного коду застосування

Це означає, що ми нарешті повністю готові до введення тексту нашої першої програми на мові С++. Традиційно, перша програма на будь-якій мові програмування виводить привітання на кшталт Hello World!. Тому, у вікні редактора програмного коду слід набрати такий текст програми:

```
#include <stdio.h>
int main()
{
     printf("Hello World!\n");
     getch();
     return 0;
}
```

Зверніть увагу в процесі набору тексту програми на такі основні особливості редактора програмного коду MS VC++:

- відображення символу * на вкладці файлу програмного коду (поряд з іменем файлу) при додаванні нового тексту (що є ознакою того, шо новий фрагмент тексту ще не збережений у файлі);
- автоматична обробка відступів (робить програмний код читабельнішим і більш зрозумілим);
- спливаючі підказки технології **IntelliSense**, які дозволяють дописати оператори мови без набору символів з клавіатури (звільняє від необхідності пошуку або запам'ятовування службових слів мови, викликів функцій і т.д.);
- автоматичне групування коду в блоки з можливістю згортання/розгортання цілих фрагментів вихідного тексту;
- виділення різним кольором окремих слів тексту програми, що мають різне призначення, а також цілих фрагментів тексту.

Після завершення набору тексту програми настійливо рекомендується зберегти його у файлі програмного коду (збережені рядки тексту програми відзначаються вертикальною зеленою лінією на початку рядка). Для цього (як і для більшості інших дій в середовищі MS VC++) існує декілька альтернативних способів: відповідна кнопка на стандартній панелі інструментів (яка має вигляд такий самий як у більшості Windows-програм), команда головного меню **File->Save**, комбінація "гарячих" клавіш Ctrl+S. Несвоєчасне збереження вихідних текстів програми часто завершується їх втратою у разі непередбачених програмних збоїв (це ж все таки Windows!) з подальшою необхідністю їх повторного набору (оскільки такі ситуації не можуть бути підставою для звільнення від здачі та захисту лабораторної роботи).

Для того, щоб програма "запрацювала" її потрібно *скомпілювати*, тобто перевести текст програми в, так звані, машинні коди. Цей процес компіляції здійснюється спеціальною програмою, яка називається *компілятором*, і яка запускається з IDE MS VC++ або натиснення функціональної клавіші **F7**, або за допомогою команди головного меню **Debug->Build Solution**. Виконавши цю дію, ми отримаємо ситуацію, яка зображена на рис. 9, і яка означає, що процес компіляції завершився невдало, оскільки компілятор виявив у нашій програмі синтаксичну помилку("error C3861: 'getch': identifier not found"). Ця помилка означає, шо компілятор не "впізнав" функцію getch(). З якої причини це сталося? Програми на С++ складаються власне як з операторів самої мови, так і викликів функцій з стандартної бібліотеки функцій мови С++ . ***getch()*** є прикладом такої функції, вона призначена для зчитування символів з консолі введення (клавіатури). У ранніх версіях бібліотеки стандартних функцій С++ ця функція мала саме такий вигляд, але в новішій версії вона була змінена на ***_getch()***. Тому для того, щоб зробити це виправлення, у вікні **Output** слід двічі клікнути на рядку, який містить повідомлення про цю помилку, після чого середовище автоматично переведе курсор у вікні редактора програмного коду у той рядок програми, який містить помилку (як показано на рис.9). Тепер можна зробити необхідне виправлення, після чого виділимо мишкою функцію ***_getch()*** і натиснемо клавішу **F1**. Відкриється вікно допомоги (фрагмент якого зображено на рис.10), в якому можна знайти всю необхідну інформацію про цю функцію. Зокрема, ми бачимо (у розділі **Requirements**), що дана

функція описана в, так званому, *header-файлі* з іменем *conio.h*. Цей файл треба підключити до нашої програми за допомогою спеціальної *директиви препроцесора* C, яка називається ***#include***. Приклад використання цієї директиви вже є у нашій програмі - перший рядок тексту програми - тому після нього слід додати новий рядок (клікаємо мишкою в самому кінці першого рядка і натискаємо клавішу **Enter**), в якому вводимо: `#include <conio.h>`

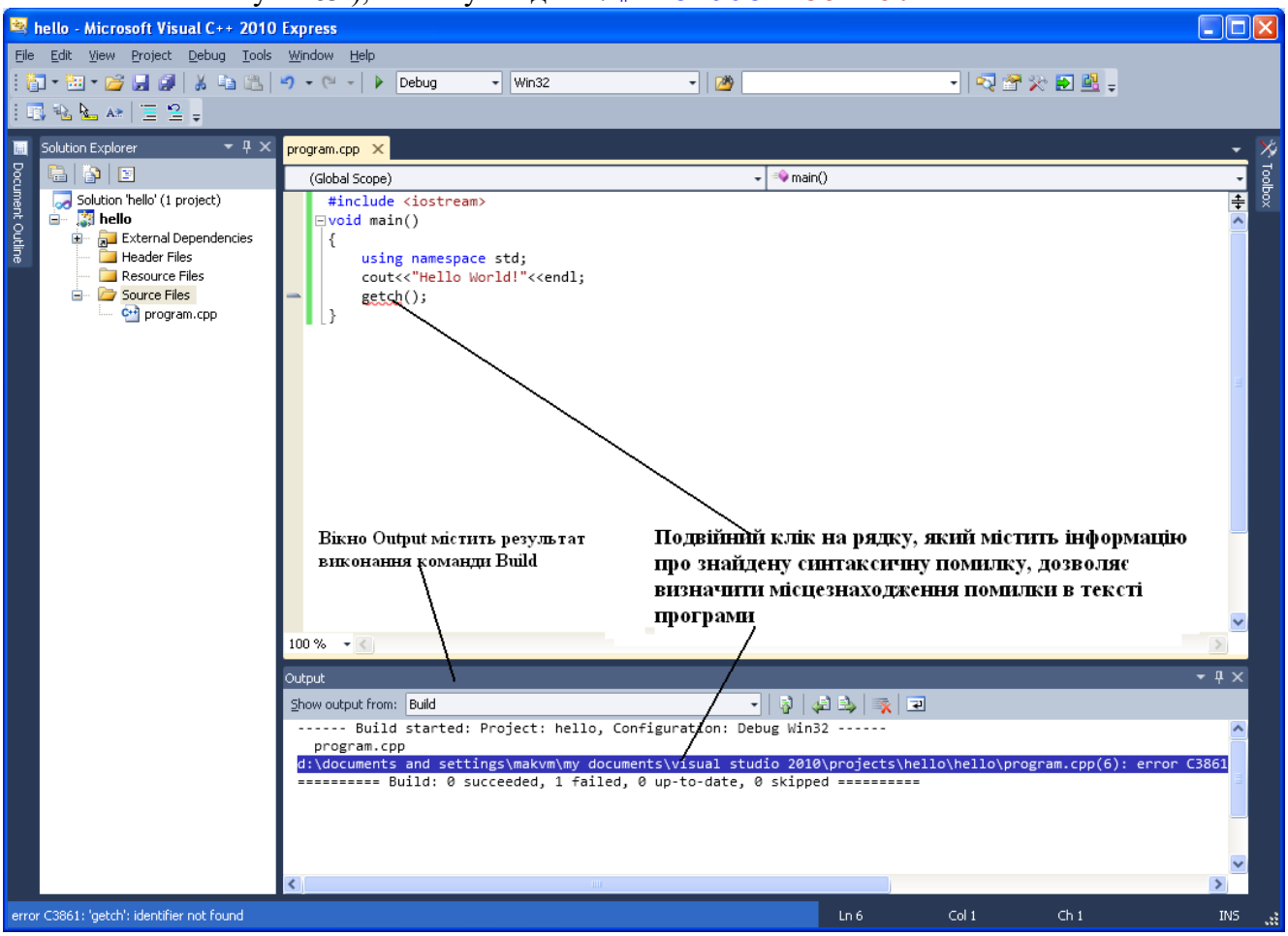
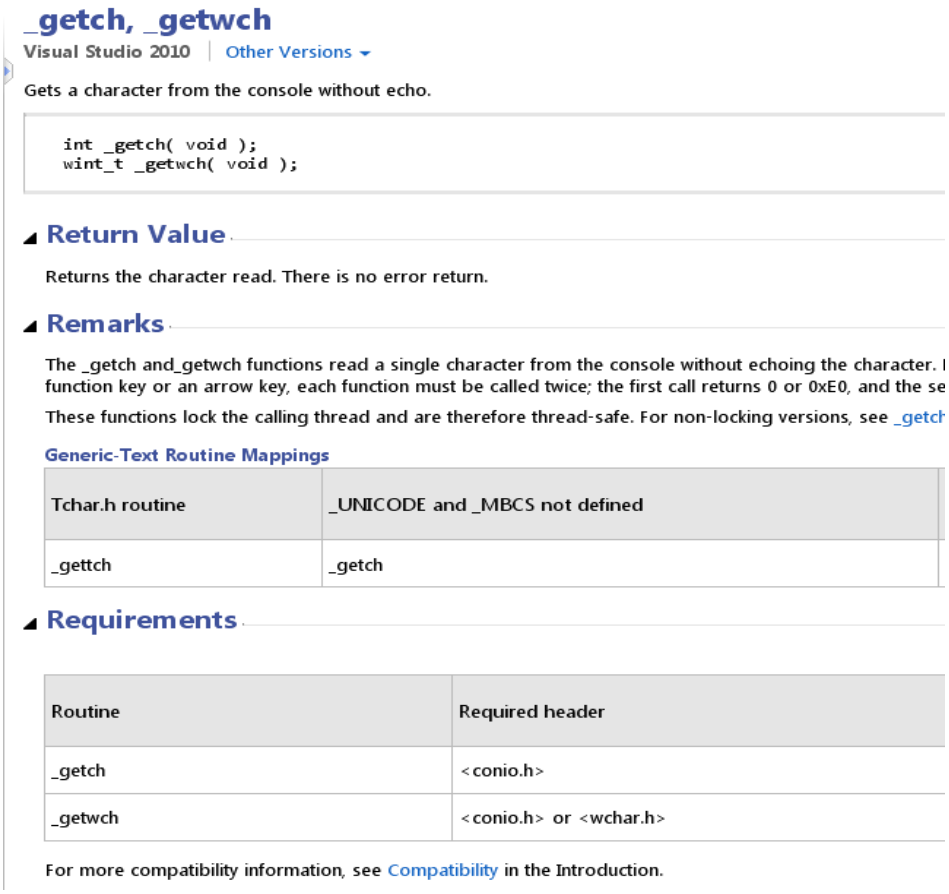


Рис.9. Результат невдалої компіляції



_getch, _getwch

Visual Studio 2010 | Other Versions

Gets a character from the console without echo.

```
int _getch( void );
wint_t _getwch( void );
```

Return Value

Returns the character read. There is no error return.

Remarks

The _getch and_getwch functions read a single character from the console without echoing the character. I
function key or an arrow key, each function must be called twice; the first call returns 0 or 0xE0, and the se

These functions lock the calling thread and are therefore thread-safe. For non-locking versions, see _getch

Generic-Text Routine Mappings

Tchar.h routine	_UNICODE and _MBCS not defined
_gettch	_getch

Requirements

Routine	Required header
_getch	<conio.h>
_getwch	<conio.h> or <wchar.h>

For more compatibility information, see Compatibility in the Introduction.

Рис.10. Вікно допомоги, в якому відображена інформація про функцію _getch()

В результаті, остаточна версія нашої першої програми на мові С++ повинна мати такий вигляд:

```
#include <stdio.h>
#include <conio.h>

int main()
{
      printf("Hello World!\n");
      _getch();
      return 0;
}
```

Знову ж таки, зверніть увагу на те, що в процесі редагування тексту програми нові рядки та рядки, які змінюються позначаються у вікні редактора програмного коду жовтою вертикальною лінією на початку рядка.

Після підключення файлу з описом функції _getch() знову запускаємо компілятор. Тепер компіляція тексту програми має завершитися успішно, після чого автоматично почне роботу інша спеціальна програма, яка називається *редактором зв'язків* або просто **Builder**. Ця програма призначена для остаточного збирання програми з різних частин, таких як відкомпільований об'єктний код, функцій стандартної (або якоїсь іншої) бібліотеки, які використовуються в тексті програмі, об'єкти різних бібліотек класів та інших. Справа в тому, що якщо в програмі використовуються функції чи об'єкти, які знаходяться не в самій програмі, а у деяких зовнішніх бібліотеках (в нашому випадку це функція _getch() та об'єкт ***cout***), то компілятор такі функції (об'єкти) не включає в тіло відкомпільованої програми, а тільки задає деяке посилання, яке вказує, що деякий програмний код, що відповідає цій функції (об'єкту), в цьому місці треба підключити. І саме розв'язанням цих посилань, тобто пошуком цих функцій (об'єктів) та вставкою їх програмного коду у відкомпільований об'єктний код програми, і займається Builder. Результатом роботи редактора зв'язків є повністю готовий до запуску виконавчий код програми (файл з розширенням .exe), який можна запустити на виконання або

за допомогою команди головного меню **Debug->Start Debugging**, або через відповідну кнопку (у вигляді зеленого наконечника вправо) з стандартної панелі інструментів, або за допомогою клавіші **F5**. Після запуску програми автоматично відкриється стандартне консольне вікно Windows, через яке програма взаємодіятиме з користувачем. У нашому випадку результат виконання програми зображено на рис.11.

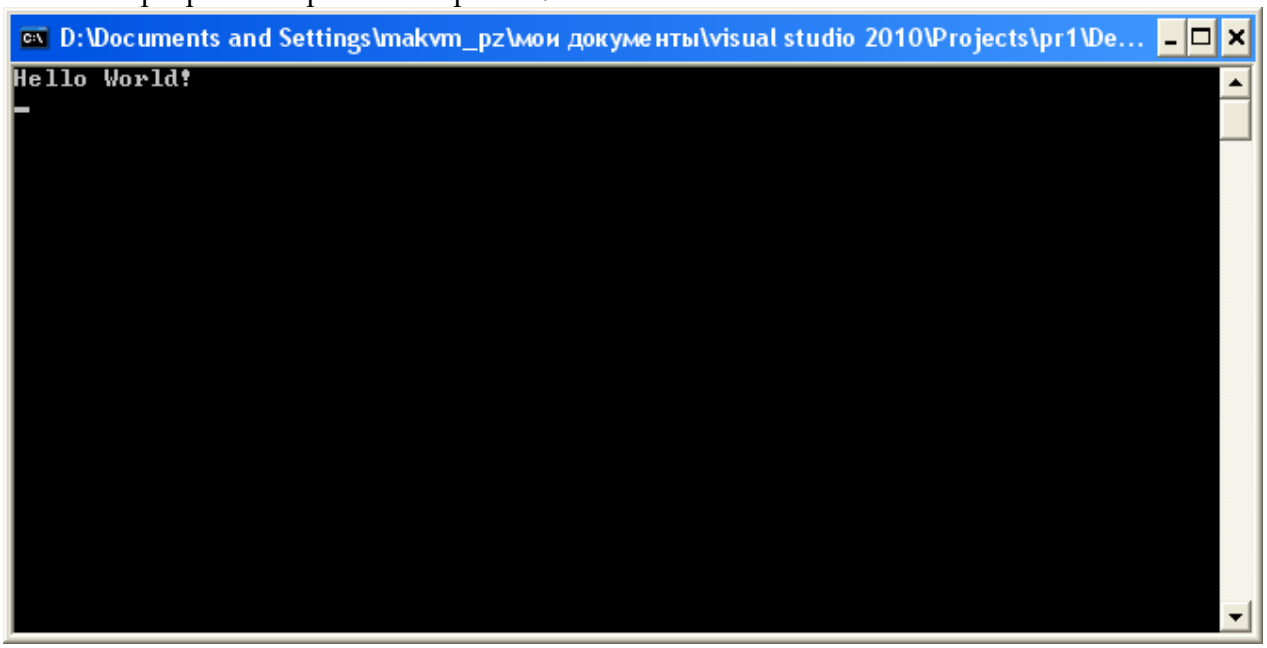


Рис.11. Результат виконання консольного застосування

Зауважимо, що консольне вікно не закривається автоматично після виведення повідомлення “Hello World!”. Така поведінка програми є наслідком використання функції _getch(), яка “призупиняє” виконання програми і змушує її чекати на натиснення користувачем довільної клавіші. Це потрібно для того, щоб користувач мав можливість побачити на екрані результат виконання програми. Таким чином, щоб закрити вікно нашої консольної програми потрібно натиснути будь-яку клавішу на клавіатурі. Поява самого повідомлення “Hello World!” у консольному вікні є результатом виконання функції виведення *printf*.

3. КОНТРОЛЬНІ ЗАПИТАННЯ

1. Для чого призначена стандартна панель інструментів? Де вона знаходиться?
2. Яка інформація відображається на стартовій сторінці? Що потрібно зробити для того, щоб стартова сторінка не відображалася при запуску середовища MS VC++?
3. У вигляді яких конструкцій організована розробка застосувань в MS VC++? Як вони між собою пов’язані і де відображається їх структура?
4. Що така консольне застосування? Які його особливості?
5. Опишіть послідовність дій, які необхідно виконати для створення нового консольного застосування.
6. Які особливості редактора програмного коду MS VC++ Ви знаєте?
7. Які є способи компіляції програми в MS VC++?
8. Як запустити програму на виконання?
9. Як відкрити раніше створений проект?

4. ЛАБОРАТОРНЕ ЗАВДАННЯ

1. Прослухати інформацію про порядок роботи в комп’ютерному класі. Проаналізувати інформацію на робочому столі Вашого ПК, занотувати в зошит порядок роботи (включення, ввід паролю, запуск середовища MS VC++, виключення).
2. Створити в текстовому редакторі файл з вихідним кодом програми, поданим вище.

3. За допомогою командного рядка скомпілювати створений файл та запустити отриману програму на виконання.
4. Відкрити середовище MS VC++, виконати всі необхідні дії для введення та запуску такої програми:

```
#include <stdio.h>
#include <conio.h>

int main()
{
    char name[15];
    printf("Enter your name: ");
    scanf("%s", name)
    printf("Hello ""%s""! You are welcome to C world\r\n", name);
    _getch();
}
```

5. Виправити у цій програмі помилку знайдену компілятором. Запустити її на виконнання, отримати та проаналізувати результат.
6. Підготувати та здати звіт про виконання лабораторної роботи.

5. СПИСОК ЛІТЕРАТУРИ

1. Прата С. Язык программирования С. Лекции и упражнения. – М. - Издательский дом Вильямс, 2014
2. Керниган Б., Ритчи Д. Язык программирования С. - М. - Финансы и статистика. - 1992. – 272 с.
3. Уэйт М., Прата С., Мартин Д. Язык С. Руководство для начинающих. - М. - Мир. - 1988. – 512 с.
4. К. Джамса. Учимся программировать на языке С++. М.: Мир, 1997. – 320 с
5. Герберт Шилдт. Полный справочник по С++. М. – С.-П.-К., Вильямс. – 2003. – 800 с.
6. Демидович Е. М. Основы алгоритмизации и программирования. Язык Си. (Учебное пособие). – Санкт-Петербург: “БХВ Петербург”. – 2006. – 439 с.