

LUCRAREA NR. 1

Prezentarea mediului de dezvoltare integrat Microsoft Visual C++

1. Obiectivele lucrării

- Familiarizarea cu mediul de dezvoltare integrat (Integrated Development Environment - IDE) Microsoft Visual C++ 2010 Professional.
- Compilarea și execuția unui program.
- Depanarea unui program.

2. Mediul de dezvoltare integrat Microsoft Visual C++ 2010

Mediul de dezvoltare integrat Microsoft Visual C++ conține instrumente pentru dezvoltarea și depanarea aplicațiilor C++:

- **editor** (pentru scrierea și modificarea programelor C++). De asemenea, editorul oferă diferite facilități, cum ar fi: copiere și lipire, recunoașterea automată a cuvintelor limbajului C++ și colorarea acestora în funcție de semnificația lor.
- **compilator** (pentru conversia codului sursă C++ în cod obiect, detectarea și semnalarea erorilor din cadrul procesului de compilare). În urma compilării pentru fiecare fișier sursă este creat un fișier obiect cu extensia .obj.
- **depanator** (pentru execuția pas cu pas a programelor, ajutând utilizatorul să găsească erorile de execuție). Mediul conține multe butoane, meniuri și alte elemente de interfață grafică cu utilizatorul (graphical user interface –GUI) ce pot fi folosite în timpul editării, compilării și depanării aplicațiilor C++.
- **editor de resurse**. Resursele reprezintă elemente de interfață cu utilizatorul, cum ar fi meniurile, cursoarele, cutiile de dialog, icon-uri, etc. Fiecare proiect construit în Visual C++ va conține un fișier cu extensia .rc ce cuprinde descrierea textuală a tuturor obiectelor de interfață care fac parte din aplicație.
- **editorul de legături** (linker) - construiește fișiere "executabile" (având extensia .exe, .dll, .lib, .ocx) folosind fișierele obiect realizate de către compilator pentru fiecare fișier din proiect.
- **generator de aplicații-șablon**, numit AppWizard, care crează "scheletul" unei aplicații Windows generice. Fișierele și conținutul acestora vor fi generate în urma unui dialog direct cu programatorul. Codul generat de către AppWizard este un cod care ajută programatorul să realizeze rapid o aplicație inițială pe baza căreia va dezvolta proiectul în continuare.

- **ClassWizard** - instrument care automatizează anumite faze ale procesului de implementare a aplicației. Dacă se dorește crearea unei noi clase sau a unei funcții prin care o anumită clasă să răspundă la apariția unui mesaj, ClassWizard va micșora efortul de programare prin generarea declarației și definiției clasei sau funcției precum și a conexiunii dintre funcție și mesaj.

Figura 1 prezintă într-o manieră grafică etapele procesului de execuție a programului în C++.

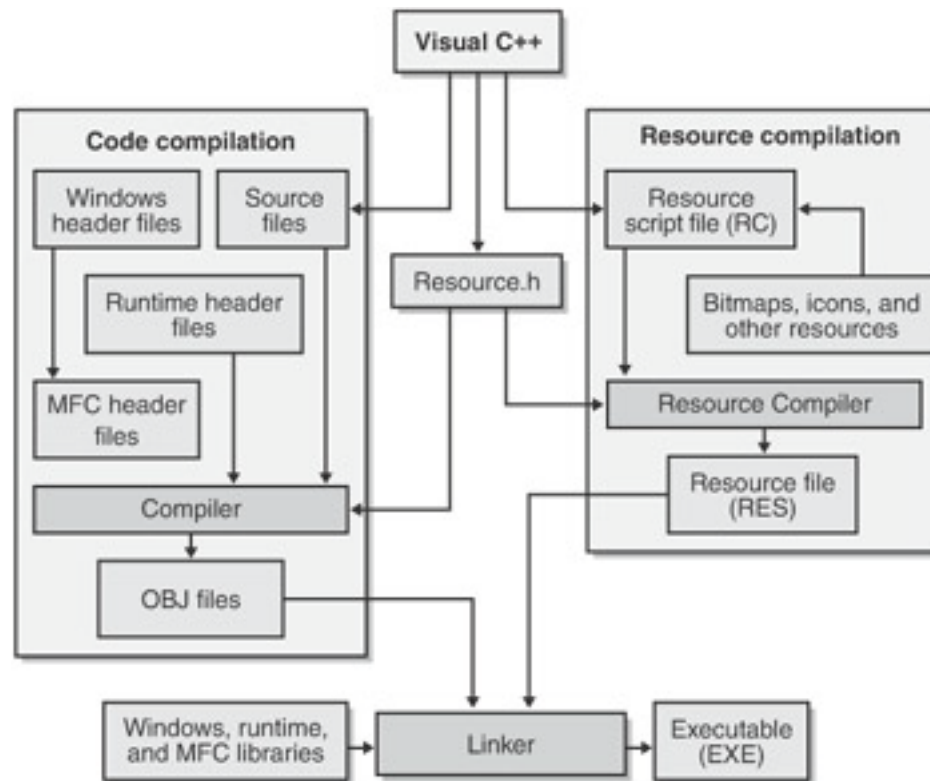




Figura 1. Etapele procesului de compilare și execuție în Visual C++

Detalii despre limbajele suportate de Microsoft Visual Studio: C/C++, VB.NET, C#, F#, precum și informații despre variantele disponibile: Express, Professional, Premium, Ultimate, Test Professional se găsesc la următorul link: http://en.wikipedia.org/wiki/Microsoft_Visual_Studio.

În octombrie 2013 s-a lansat Microsoft Visual Studio 2013, împreună cu platforma.NET 4.5.1.

Mediul de dezvoltare integrat Microsoft Visual C++ poate fi accesat apăsând butonul Start  și selectând All Programs → Microsoft Visual Studio 2010  → Microsoft Visual Studio 2010. Se va deschide o fereastră ca în Figura 2.

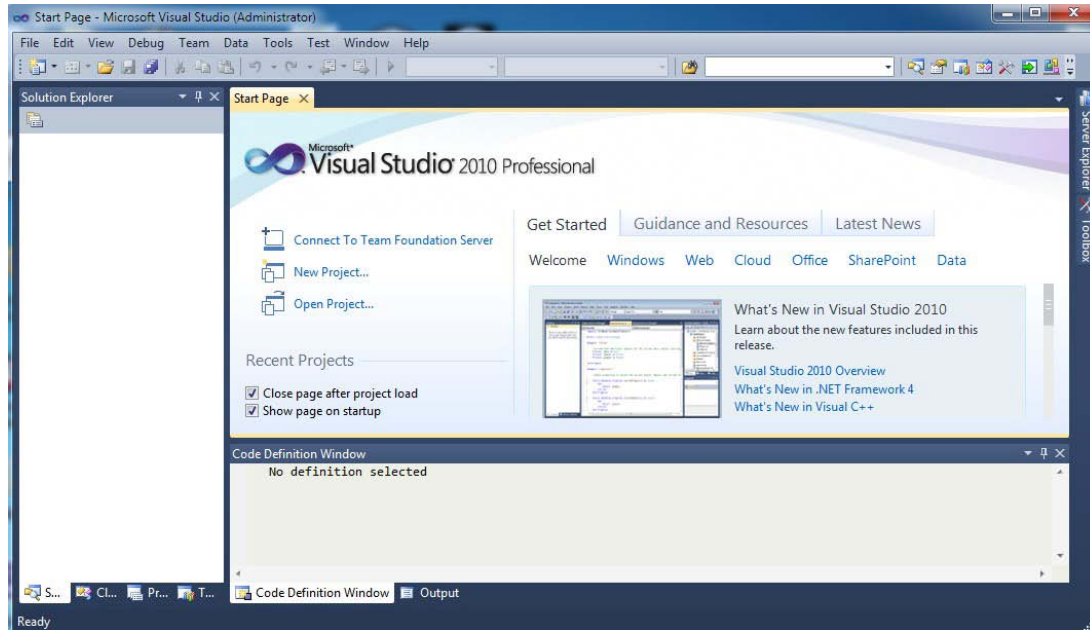


Figura 2. Mediul Visual Studio 2010 Professional

În această fereastră identificăm următoarele zone: zona din stânga conține fereastra Solution Explorer, în zona din dreapta este fereastra de editare care inițial arată pagina de Start, iar în zona de jos se găsește un tab pentru fereastra Output. Fereastra Solution Explorer permite gestionarea fișierelor din cadrul unei soluții: navigarea în lista de fișiere, afișarea conținutului acestora în fereastra de editare și adăugarea fișierelor noi în soluția curentă. *Soluția* reprezintă o colecție de unul sau mai multe proiecte înrudite între ele, precum și alte resurse utilizate în crearea aplicației. Un *proiect* reprezintă o colecție de fișiere care în urma compilării și rulării generează un program executabil.

Informațiile detaliate despre un proiect sunt stocate într-un fișier XML cu extensia .vcxproj în directorul proiectului. Informațiile despre proiectele unei soluții sunt stocate în două fișiere cu extensia .sln, respectiv .suo. Soluția se creează automat atunci când se creează un proiect, cu excepția situației în care se adaugă un proiect la o soluție deja existentă.

Principalele tipuri de proiecte sunt:

- Console Application – aplicație care poate rula în mod consolă (din linia de comandă), fără interfață grafică;
- Windows Application – aplicație executabilă în format Windows și care reprezintă standardul de aplicații care lucrează sub sistemul de operare Microsoft Windows.
- Class Library – bibliotecă de clase, fișier bibliotecă cu extensia .dll.

3. Crearea unei aplicații în mod consolă (Win32 Console Application)

Se selectează 'New' -> "Project" din meniul "File". Pe ecran va apare o fereastră de dialog ca în Figura 3.

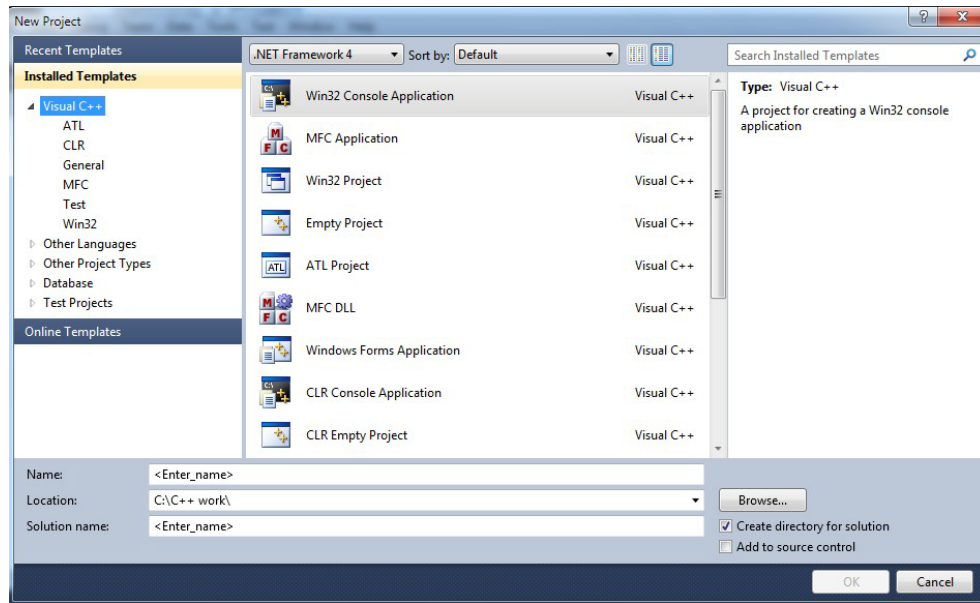


Figura 3. Crearea unui nou proiect în Visual Studio 2010 Professional

Din cadrul panoului Installed Template se alege “Visual C++”, iar apoi se va selecta “Win32 Console Application”. În câmpul “Name” se va scrie numele proiectului (de exemplu Lab1_p1), în câmpul “Location” se va scrie calea unde se vor stoca fișierele din cadrul proiectului, iar în câmpul “Solution name” se va scrie numele soluției (Lab1_p1), aceasta va avea implicit același nume cu cel al proiectului. După apăsarea butonului OK, se va deschide fereastra Win32 Application Wizard ca în Figura 4.

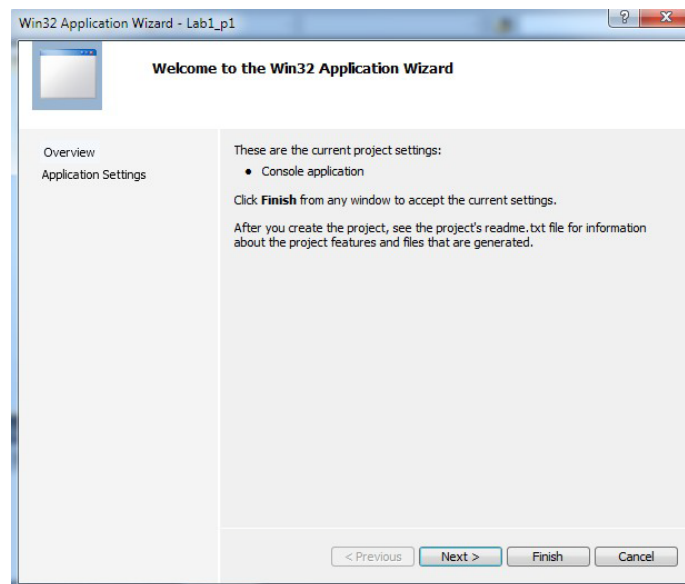


Figura 4. Win 32 Application Wizard pentru crearea unei aplicații de tip consolă

Această fereastră afișează setările curente ale proiectului, iar după apăsarea butonului Finish, wizard-ul va crea toate fișierele proiectului pe baza acestor setări. Astfel, dacă se apasă Application Settings, se pot alege opțiunile dorite pentru proiectul curent, ca în Figura 5.

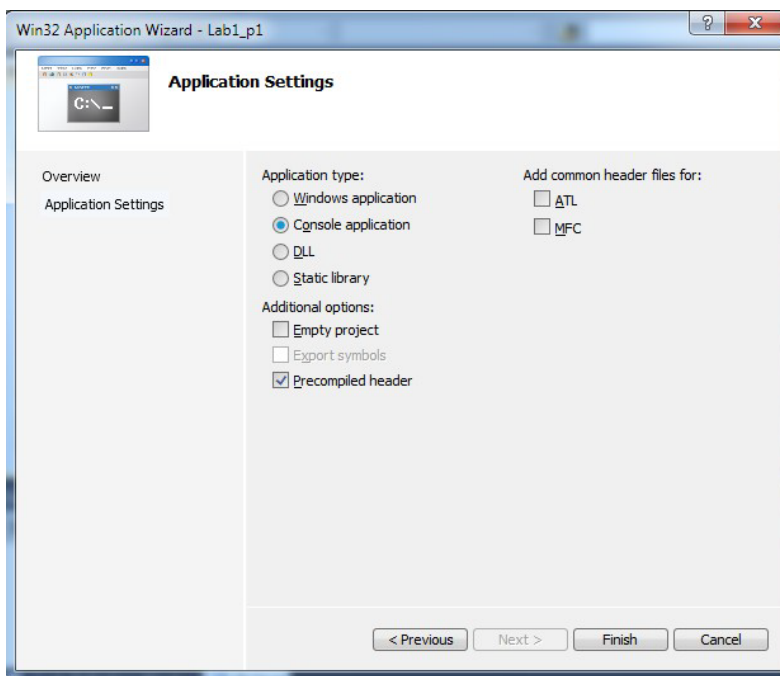


Figura 5. Fereastra pentru selectarea tipului de proiect

Proiectul creat se va deschide automat în panoul Solution Explorer, ca în Figura 6. Conținutul fiecărui fișier din cadrul proiectului poate fi vizualizat în fereastra de editare dacă se apasă dublu click pe numele fișierului.

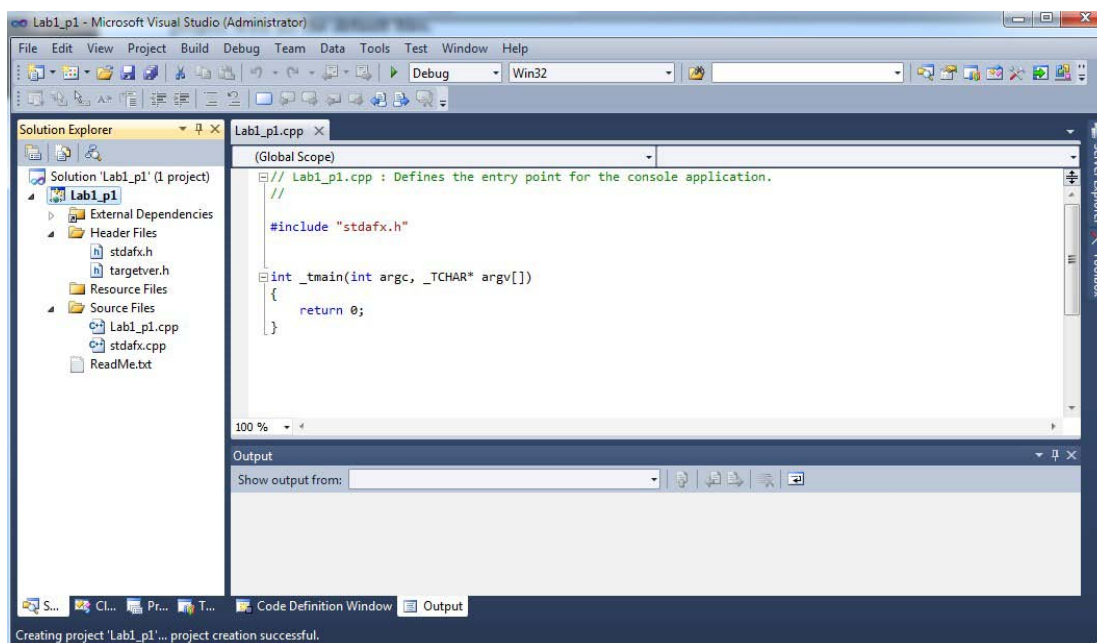


Figura 6. Fereastra Solution Explorer

Obs. În partea de jos a ferestrei se observă următoarele tab-uri: **Class View**, **Property Manager** și **Team Explorer**. Tab-ul Class View afișează clasele, membrii claselor și funcțiile din cadrul proiectului, iar în cazul în care aplicația nu conține clase, nu se va afișa nimic. Tab-ul Property Manager arată proprietățile care au fost

stabilite pentru versiunile Debug și Release ale proiectului. Aceste proprietăți pot fi modificate apăsând click dreapta pe o proprietate dorită și selectând Properties din meniul afișat.

Obs. Visual C++ 2010 permite construirea aplicațiilor în două configurații: Debug și Release. Versiunea Debug include informații suplimentare care ajută la depanarea programului, astfel se pot urmări valorile variabilelor în timpul execuției, folosirea punctelor de întrerupere, execuția programului pas cu pas, etc. Versiunea Release nu include informații despre depanare, iar codul este optimizat, se execută mai rapid.

4. Editarea codului sursă

Wizard-ul Visual C++ generează un program de tip consolă Win32 care poate fi compilat și lansat în execuție. Deoarece programul nu face nimic, se vor adăuga următoarele linii de cod (marcate cu bold) în fereastra de editare a fișierului Lab1_p1.cpp:

```
#include "stdafx.h"
#include <iostream>

int _tmain(int argc, _TCHAR* argv[])
{
    std::cout << " Bine ati venit in lumea programelor C++!\n";
    return 0;
}
```

Obs. Pentru a numerota liniile de cod se selectează din meniul "Tools" -> "Options..." , iar din fereastra de dialog deschisă se extinde meniul Text Editor și se selectează opțiunea C/C++. Se bifează căsuța "Line Numbers" din panoul corespunzător secțiunii General și se apasă butonul OK.

Obs. `_t` din fața numelui funcției main, precum și a parametrului CHAR reprezintă o convenție Microsoft pentru a permite lucrul cu bibliotecile Unicode. Dacă macro-ul UNICODE este definit, `_t` înseamnă "wide character" , dacă UNICODE nu este definit atunci `_t` înseamnă "single byte character".

Exemple:

Macro	UNICODE	non-UNICODE
<code>_tmain</code>	<code>wmain</code>	<code>main</code>
<code>_TCHAR</code>	<code>wchar_t</code>	<code>char</code>
<code>_T("Hi")</code>	<code>L"Hi"</code>	<code>"Hi"</code>
<code>LPTSTR</code>	<code>wchar_t *</code>	<code>char *</code>

Implicit, proiectul are setată opțiunea de utilizare a bibliotecilor Unicode. Pentru a modifica această opțiune se selectează din meniul Project -> Properties (Alt+F7) și de la secțiunea General se selectează proprietatea Character Set și se alege Not Set.

După scrierea codului sursă se salvează fișierul folosind combinația de taste <Ctrl-S> sau selectând din bara de meniuri "File" -> "Save".

5. Compilarea și execuția programului

Pentru compilarea fișierului sursă C++ se selectează comanda **Compile** (Ctrl+F7) din meniul **Build**. Mesajele compilatorului și erorile vor apare în fereastra Output. Dacă nu sunt erori, după compilare va apare mesajul "Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped" (Figura 7). În cazul în care sunt erori, după corectarea acestora fișierul trebuie recompilat.

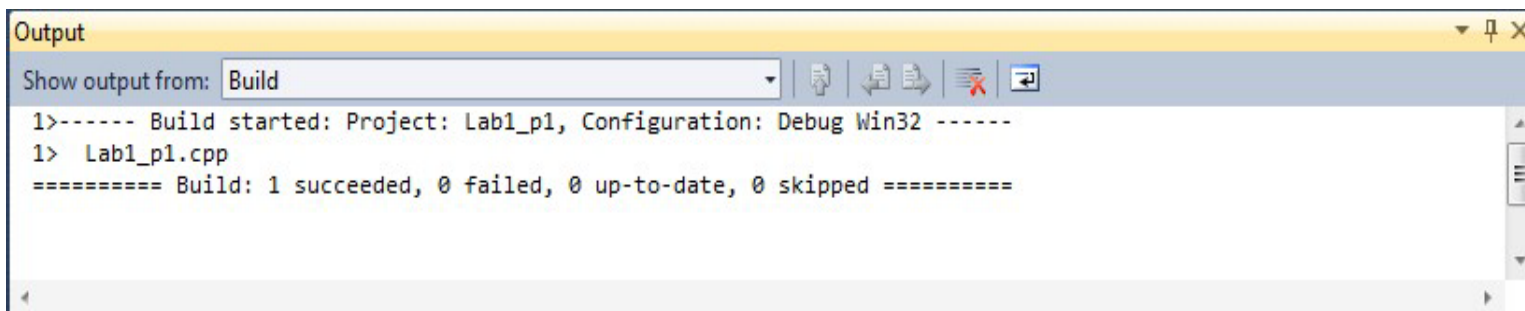


Figura 7. Rezultatul compilării programului

Compilarea și editarea de legături se pot face prin selectarea comenzii **Build Solution** (F7) din meniul **Build**. Pentru execuția programului se selectează comanda **Start Without Debugging** (Ctrl+F5) din meniul **Debug**. Se va deschide fereastra din Figura 8.

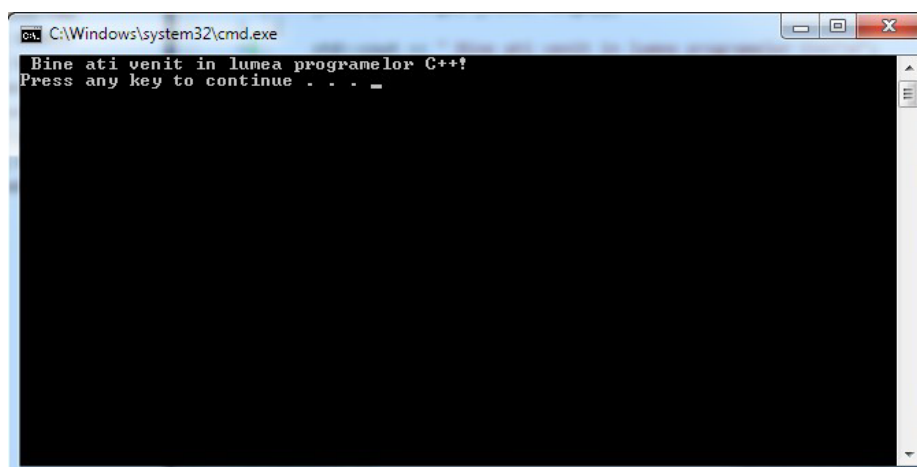


Figura 8. Fereastra cu execuția programului

6. Depanarea unui program

Procesul de depanare (debug) permite execuția programului în mai multe moduri, astfel încât să poată fi urmărite rezultatele unor funcții și valorile unor variabile în decursul execuției programului. Principalele modalități de depanare sunt:

- **Start Debugging** (F5). Această comandă începe procesul de depanare.
- **Step Into** (F11) din meniul Debug. Debugger-ul va executa linia curentă și dacă este o funcție, va sări la funcția respectivă și va începe să execute liniile acelei funcții pas cu pas.
- **Step Over** (F10). Efectul este similar cu Step Into, numai că la întâlnirea unei funcții nu va mai intra în codul funcției respective, ci va întoarce doar rezultatul execuției funcției.
- **Step Out** (Shift+F11). Dacă debugger-ul a intrat în codul unei funcții și se lansează această comandă, se va parcurge funcția automat până la sfârșit și se va întoarce pe un nivel mai sus.

O modalitate de întrerupere a execuției unui program într-un anumit punct definit de utilizator, cu posibilitatea continuării ulterioare a execuției este folosirea unui breakpoint. Un breakpoint se inserează în codul sursă prin selectarea liniei în care se dorește întreruperea programului și selectare comenzii **Toggle Breakpoint** din meniul **Debug** sau prin acționarea tastei F9. Pentru a utiliza breakpoint-ul nou creat, se folosește comanda **Start Debugging**. Programul se va executa până va întâlni breakpoint-ul, moment în care se va opri și va aștepta comenzile ulterioare ale utilizatorului. Se pot utiliza aici comenzile Step Into, Step Over, Step Out. Lista cu breakpoint-uri poate fi vizualizată folosind **Breakpoints** din meniul **Debug -> Window** sau combinația de taste Alt+F9.

Pentru închiderea soluției curente se folosește comanda **Close Solution** din meniul **File**.

7. Crearea unei aplicații Windows (MFC Application)

Microsoft Foundation Classes (MFC) este o bibliotecă de clase C++, dezvoltată de firma Microsoft ce încapsulează funcțiile Windows API formând astfel baza pentru aplicațiile care rulează sub sistemul de operare Microsoft Windows. Windows API (Application Programming Interface) este o interfață pentru programarea aplicațiilor specifice sistemului de operare.

Pentru crearea unei aplicații de tip Windows se selectează "New"->"Project" din meniul "File". Din cadrul panoului Installed Template se alege "Visual C++" -> "MFC", iar apoi se va selecta "MFC Application".

Se scrie numele proiectului, al soluției și apoi se apasă OK. Se va deschide wizard-ul aplicației MFC ca în Figura 9. Acesta va genera automat scheletul programului.

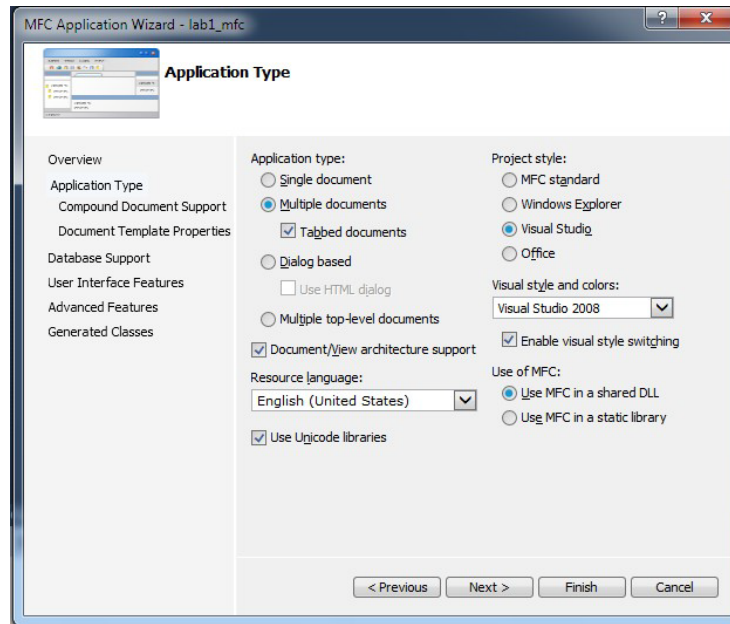


Figura 9. Fereastra MFC Application Wizard

Se debifează opțiunea Tabbed documents și se selectează Windows Native/Default din lista “Visual style and colors”. Se selectează Advanced Features și se debifează următoarele opțiuni: Explorer docking pane, Output docking pane, Active X controls, Common Control Manifest., astfel încât fereastra va arăta ca în Figura 10.

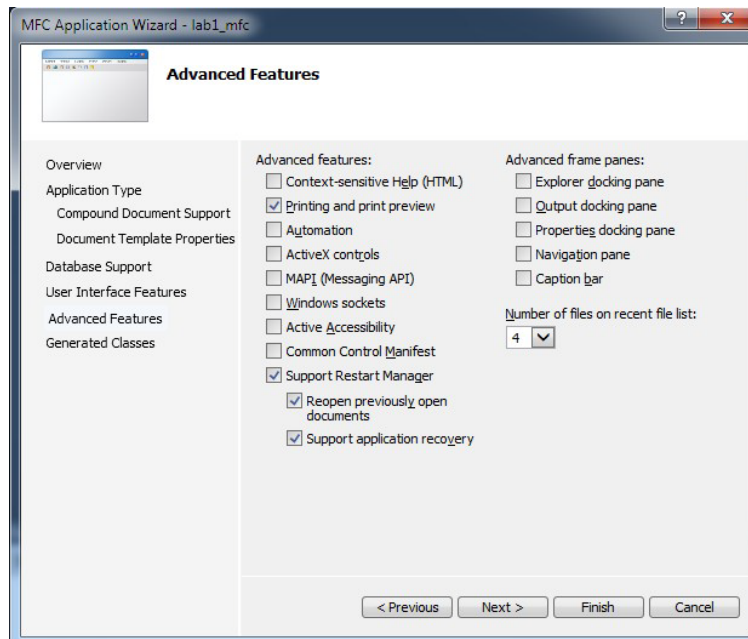


Figura 10. Fereastra Advanced Features

Se apasă butonul Finish pentru crearea proiectului. În Solution Explorer se pot vizualiza fișierele sursă create și câteva fișiere de resurse, ca în Figura 11. Astfel există fișiere cu extensia .cpp ce conțin codul sursă C++, fișierele .h ce reprezintă fișierele antet, precum și fișiere .ico ce conțin imagini. Există de asemenea și un fișier ReadMe.txt care conține detalii despre fișierele create de wizard-ul MFC Application.

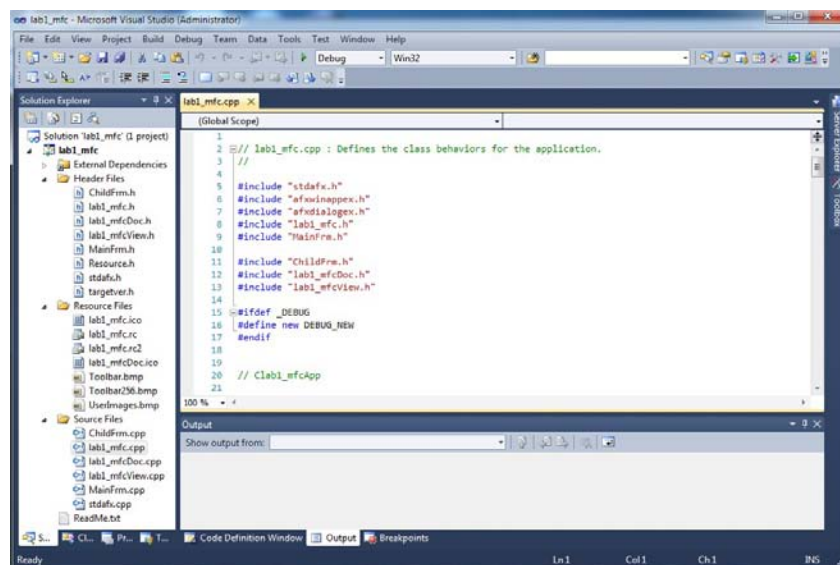


Figura 11. Fereastra Solution Explorer

În urma compilării și execuției programului, se va afișa o fereastră cu meniuri și bară de instrumente, ca în Figura 12.

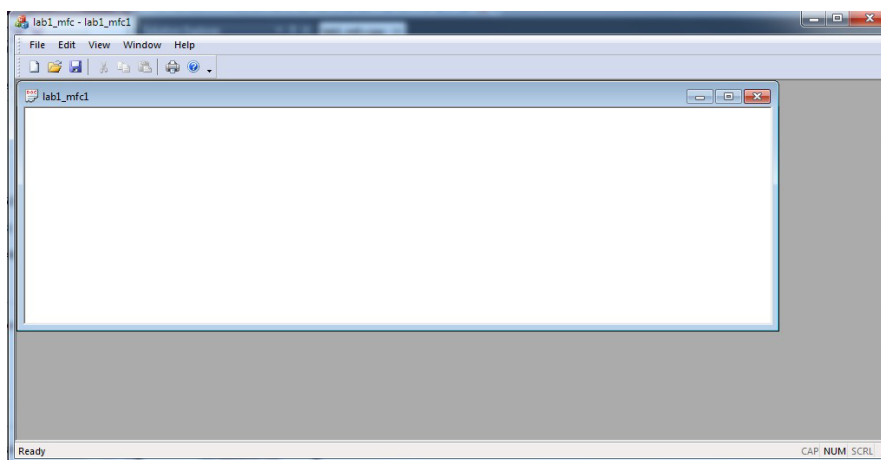


Figura 12. Fereastra rezultată în urma execuției programului

Se observă că meniurile sunt funcționale, se pot verifica următoarele operații: deschiderea unui document nou, salvarea pe disc a acestuia etc.

8. Verificarea cunoștințelor

- Ce diferență este între un proiect și o soluție în Visual C++?
- Care sunt pașii ce trebuie parcurși în realizarea unei aplicații C++?
- Scrieți un program care calculează suma și produsul a N termeni citați de la consolă. Depanați pas cu pas programul, urmărind valorile variabilelor în timpul execuției.