مجلة جامعة بابل / العلوم الصرفة والتطبيقية / العدد (١) / المجلد (٢٤) : ٢٠١٦

تصميم وبناء بيئة برمجية متآلفة مع بيئة نظام Windows

علي هادي حسن محمد كامل حسن عوض

كلية العلوم- جامعة بابل

الخلاصة

يهدف هذا البحث إلى تصميم وبناء بيئة برمجية خاصة لتحرير وتجميع وتنفيذ برامج لغة (Turbo Assembly 5.0) من شركة (Borland) على بيئة متآلفة مع بيئة نظام التشغيل (Windows) .

لقد تم تنفيذ البيئة البرمجية التي سميت (Babylon)، من خلال تصميم وبناء محرر يستخدم بعض الأدوات الضرورية والمتميزة التي تستخدمها تقنية (IntelliSense) المبنية من قبل شركة (Microsoft) في تطبيقات (Visual Studio). ثم تصميم وبناء البرامج التي تربط المجمّع (Borland Turbo Assemble) مع الملفات المصدرية المتولدة من المحرر وإنتاج ملفات (Object Files) و(List Files) المطلوبة. وأخيراً تصميم وبناء برامج التي تربط البرنامج الرابط (linker) مع ملفات الهدف (Object Files) لتوليد الملفات التنفيذية (Execution Files).

إن البيئة البرمجية التي تم تصميمها وتنفيذها هي بيئة فعالة و كفوءة وسهلة الاستعمال تعطي للمبرمج تسهيلات ومساعدات في تحرير البرامج وتنفيذها وكذلك تزوده ببعض الأدوات التي يحتاجها أثناء التعامل مع لغة (Assembly).

الكلمات المفتاحية: محرر لغة الاسمبلي، برامج للتجمع والربط لغة الاسمبلي.

Abstract

This research aims at designing and building a software environment to edit, compiled and implement assembly programs to improve Turbo Assembly 5.0 from the Borland Company to work on a harmonious environment with Windows operating system.

The software environment called (Babylon) has been implemented by designing and constructing an Editor that uses some necessary and distinctive instruments used by IntelliSense built by a Microsoft in its applications Visual Studio. Then design and build the programs that connect the Borland Turbo Assembler 5.0 with the source files generated by our editor to produce the required Object Files and List Files. Finally we design and build programs that connect the program which links the linker with the Object Files to generate Execution Files.

The Babylon software environment that has been designed and implemented is efficient and easy to use and it gives the programmer facilities to assist him in editing the programs and implementing there. Moreover it provides him with some of the tools needed to deal with the Assembly language.

Key words: Assembly Editor, Software link Assembler and Linker.

١ – المقدمة

منذ ظهور الحاسوب لأول مرة، احتاج المبرمجون والمستخدمون إلى وسائل وادوات تساعدهم في تحرير البرامج والنصوص. في البداية كانت الحاجة مقتصرة على تحرير أرقام وحروف قليلة لحاجات محدودة، ولكن بعد تطور أنظمة الحاسبات التي تطورت معها البرمجيات، فكر المبرمجون بوضع برمجيات خاصة تسيطر وتدير عمل هذه الحواسيب فظهر لأول مرة مفهوم أنظمة التشغيل (Operating Systems).

أن برامج التحرير كانت في بداية الأمر هي جزء من أنظمة التشغيل، لذلك كان تطور برامج التحرير تسير بنفس الوقت الذي فيه أنظمة التشغيل، وزاد الإلحاح عليها بازدياد تطور لغات البرمجة، ولذلك قررت بعض الشركات المنتجة للغات البرمجة بإنتاج محرر خاص لكل لغة. في البداية كانت المحررات بسيطة لا تختلف كثيراً عن أي محرر نصوص موجود في ذلك الحين، ومن أمثلة هذه المحررات: محرر لغة (GW-Basic).

ومع تطور اللغات ودخول مفاهيم جديدة عليها كالبرمجة الكيانية (Object Oriented Programming)، أصبحت هذه المحررات غير مجدية مما حدى بالشركات البحث لتصميم وبناء محررات اكثر فعالية وسهولة، فظهر جيل جديد من المحررات وهو ما يسمى بيئة التطوير المتكاملة (Integrated Development Environment) أو اختصاراً (IDE)، وتسهل

هذه البيئة عمليات الترجمة (Compile) والتحرير(Editing) والتنفيذ (Running) والتنقيح (Debugging) والتي لا غنى عنها عند كتابة البرامج [ورنر فيبل، ٩٥].

يوجد العديد من المحررات والبيئات التي اشتهر تداولها مع برامجيات أنظمة الحاسبات الشخصية (PC) واللغات التطبيقات التي تعمل عليها ومن اشهرها :

أ- المحرر EdLin

كان من ضمن البرمجيات الملحقة ببعض أنظمة التشغيل؛ برامج خاصة لتحرير النصوص ومن أمثلة هذه البرامج: البرنامج (EDLIN.EXE) اختصاراً لـ(Edit Line) الذي كان مضمن مع البرمجيات الملحقة بنظام التشغيل(MS-DOS) من شركة (Microsoft)، وكان لهذا البرنامج خصائص ومواصفات جيدة حينها، ولكن هذا البرنامج كان ملائماً للملفات الصغيرة وكان غير عملي للملفات الكبيرة الحجم وبالتالي كانت عمليات التحرير مقتصرة على أجزاء من الملف أو بمعنى آخر تتم قراءة ذلك الجزء المعني من الملف وهذه هي أحد نقاط الضعف في هذا المحرر [Microsoft Corporation, 1987].

ب- المحرر Edit

ثم ظهرت بعده برامج أخرى لتحرير النصوص كالبرنامج (EDIT.EXE) الذي لا يزال المبرمجون يستخدمونه حتى الآن عند العمل مع بيئة (DOS) وذلك للخصائص المتميزة التي يمتلكها هذا المحرر، ومن هذه الخصائص هو انه يمكن للمستخدم أن يستعرض النص بالكامل، وكذلك يمكن فتح اكثر من ملف في نفس الوقت، إضافة إلى سهولة التغيير الإضافة على النص[Microsoft Corporation, 1987].

ج- بيئة التطوير المتكاملة (IDE)

وتتألف هذه البيئة من عدة عناصر أهمها: شريط القوائم (Menu Bar) وشريط الحالة (Status Bar) وصندوق الإغلاق (Close Box) وشريط التدرج (Scroll Bar) اللذان يستخدمان من خلال الفارة لإغلاق النوافذ وللتنقل ضمن الملف على الترتيب، وكذلك تمييز الكلمات المحجوزة للغة بإضاءتها أو تغيير لونها، وأيضا تقنية المساعدة الفورية (Help On-Line)، ومن افضل الأمثلة على هذه البيئة: محرر لغة (Turbo Pascal 7.0) من شركة (Borland) [ورنر فيبل، ٩٥].

د- بيئة (Microsoft Visual Studio)

واستمرت التطورات في محررات اللغات ولكنها كانت تدور حور بيئة (IDE) حتى ظهور نظام (Windows) من شركة (Microsoft) الذي يعتمد عمله بشكل أساسي على استخدام الفارة والصور، والذي تم تصميمه وبرمجته اعتماداً على أسلوب البرمجة الكيانية (Object Oriented Programming) التي كانت الكثير من اللغات لا تدعمها، ولذلك ظهرت برامج جديدة للتحرير تعمل مع نظام (Windows) وأضيفت إليها تطويرات كثيرة لكي تحاكي احتياجات المبرمجين، ومن اشهر هذه البرامج: مجموعة برامج (Visual Studio)من شركة (Microsoft) ومجموعة برامجيات مثل (Borland Delphi 5) من شركة (Borland)، التي باستخدامها يتمكن المبرمجون من إنشاء برامج أخرى تعمل مع نظام (Windows)، ومع مجيء هذا الجيل من المحررات، أدخلت تقنية جديدة إلى بيئة (IDE) وهي تقنية (IntelliSense) وهذه التقنية لها فوائد عديدة تسهل على المبرمج الكثير من التعقيدات البرمجية وخاصة عند التعامل مع البرمجة الكيانية [Microsoft Corporation; 2000].

٢- تصميم بيئة Babylon

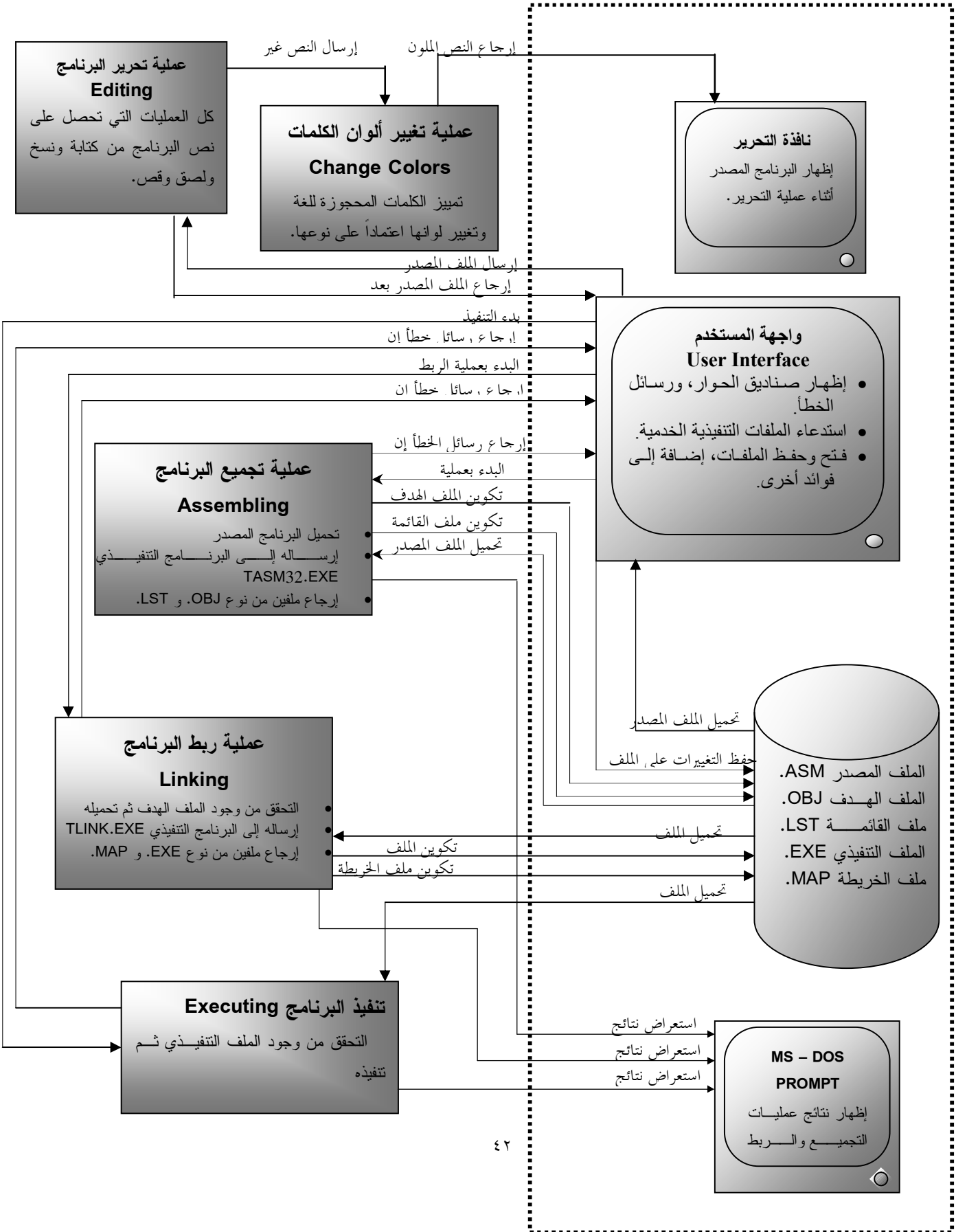
إن اغلب التقنيات والبرمجيات التي استخدمت في تصميم وبناء البيئة البرمجية(Babylon) هي تقنيات إمّا موجودة أصلاً في نظام (Windows) أو مشابهة لتقنية موجودة في هذا النظام، وقد استخدمت التقنيات التي تقدمها لغة (VisualBasic) [انيس محمد توفيق؛ ١٩٩٨ و Eric Winemiller *et al.;* 1999].

في بناء بيئة (IDE) للمحرر كشريط القوائم (Menu Bar) وصناديق الحوار (Message Boxes) وصناديق النصوص (Text Boxes)، أما الأجزاء التي ليست في احتكاك مباشر مع المستفيد كعملية تحويل الملف المصدر (Source file) إلى ملف تنفيذي (Execution file)، فيتم استدعاء برامج خاصة تعمل على بيئة (DOS) لتقديم هذه الخدمة، إضافة إلى تقنية ملفات المساعدة (Help files) التي يوفرها نظام (Windows) ضمنياً والتي أدخِلَتْ إلى محرر هذه البيئة لمساعدة المستفيد للحصول على المعلومات. هذه المعلومات عن لغة (Assembly) [Peter Abel; 1998].

أن المخطط الكتلي لبيئة البرامجيات (Babylon) الموضح في الشكل (١)، يتكون من عدة كتل (Blocks) تمثل العمليات الأساسية لهذه البيئة وهي:

٢-١ عملية تحرير البرنامج (Editing):

أن كتلة عملية تحرير البرامج هي المسؤولة عن التغييرات التي تحدث للملف المصدر، فأي عملية حذف أو إضافة أو تغيير يحدث في الملف يجب أن يمر بهذه الكتلة، ويأتي الملف المصدر من واجهة المستخدم، حيث يقوم المستخدم اما بفتح الملف المصدر المخزون على القرص، وعندئذ لا بد من تغيير ألوان هذا الملف قبل استعراضه على نافذة التحرير، لذلك يرسل إلى كتلة عملية تغيير ألوان الكلمات وبعد ذلك يتم إرساله إلى نافذة التحرير. او عند توليد ملف نصي جديد حيث يتم معالجة ألوان الكلمات لسطر واحد أثناء التحرير. وعند إكمال عملية التحرير لا بد من إرجاع الملف إلى واجهة المستخدم لغرض الخزن.


عملية تحرير البرنامج
Editing
كل العمليات التي تحصل على نص البرنامج من كتابة ونسخ ولصق وقص.
إرسال النص غير
إرجاع النص الملون
عملية تغيير ألوان الكلمات
Change Colors
تمييز الكلمات المحجوزة للغة وتغيير لوانها اعتماداً على نوعها.
نافذة التحرير
إظهار البرنامج المصدر أثناء عملية التحرير.
إرسال الملف المصدر
إرجاع الملف المصدر بعد
بدء التنفيذ
إرجاع رسائل خطأ إن
البدء بعملية الربط
إرجاع رسائل خطأ ان
واجهة المستخدم
User Interface
إظهار صناديق الحوار، ورسائل الخطأ.
استدعاء الملفات التنفيذية الخدمية.
فتح وحفظ الملفات، إضافة إلى فوائد أخرى.
إرجاع رسائل الخطأ إن
البدء بعملية
عملية تجميع البرنامج
Assembling
تحميل البرنامج المصدر
إرساله إلى البرنامج التنفيذي TASM32.EXE
إرجاع ملفين من نوع OBJ. و LST.
تكوين الملف الهدف
تكوين ملف القائمة
تحميل الملف المصدر
تحميل الملف المصدر
حفظ التغييرات على الملف
الملف المصدر ASM.
الملف الهدف OBJ.
ملف القائمة LST.
الملف التنفيذي EXE.
ملف الخريطة MAP.
عملية ربط البرنامج
Linking
التحقق من وجود الملف الهدف ثم تحميله
إرساله إلى البرنامج التنفيذي TLINK.EXE
إرجاع ملفين من نوع EXE. و MAP.
تحميل الملف
تكوين الملف
تكوين ملف الخريطة
تحميل الملف
تنفيذ البرنامج Executing
التحقق من وجود الملف التنفيذي ثم تنفيذه
استعراض نتائج
استعراض نتائج
استعراض نتائج
MS – DOS
PROMPT
إظهار نتائج عمليات التجميع والربط

الشكل (١) المخطط الكتلي للعمليات الاساسية لبيئة Babylon

٢-٢. عملية تغيير ألوان الكلمات المحجوزة (Change Colors):

في هذه الكتلة يتم تمييز كل كلمة من كلمات النص القادم من كتلة التحرير ومقارنتها مع الكلمات المحجوزة للغة والمخزونة في مصفوفة خاصة، وذلك بالبحث عن الكلمة في هذه المصفوفة ثم إرجاع اللون الخاص بنوع هذه الكلمة .

ان عملية تمييز الكلمات المحجوزة (Language Keywords) للغة Assembly والمكتوبة في الملف المصدر (Source file) من حيث كونها تعليمات (Instructions) أو توجيهات (Directives) أو معاملات (Operators) أو مسجلات (Registers) هي من أهم العمليات التي يقوم بها المحرر. فقد وضع لكل نوع من هذه الكلمات المحجوزة لون خاص يميزها عن الأنواع الأخرى، ولغرض معرفة نوع الكلمة المحررة فقد وضعت كل الكلمات المحجوزة في مصفوفة واحدة ويتم البحث عن الكلمة المحررة في هذه المصفوفة لمعرفة نوعها وتحديد لونها. لقد اخترنا خوارزمية البحث الثلاثي (Ternary Search Algorithm) لما تملكه من افضلية حيث تستغرق هذه الخوارزمية من الوقت (Log n) في أسوأ الحالات ولذلك فهي من الناحية النظرية تعتبر أسرع خوارزمية بحث في المصفوفات ذات البعد الواحد. ان فكرة هذه الخوارزمية باختصار (اختبار العنصر n/3 حيث n يمثل عدد عناصر المصفوفة، للمساواة مع قيمة معطاة ولتكن k، ثم يحتمل أن تختبر عنصر الموضع 2n/3 فإما أن يتطابق مع k أو يختزل حجم المصفوفة إلى الثلث

[Ellis Horowitz; 1997]

تعتبر عملية البحث عن كل كلمة ومقارنتها ثم إرجاع اللون المناسب لها عملية طويلة نسبياً ولذا فهي تأخذ الكثير من وقت المعالجة. بعد إنهاء المعالجة يرسل النص الملون إلى نافذة التحرير لإكمال التحرير.

٢-٣.عملية تجميع البرنامج (Assembling):

تقوم هذه الكتلة بجلب اسم الملف المصدر (Source File) الجاري تحريره ضمن نافذة التحرير النشطة، ثم التأكد من أن البرنامج قد تم خزنه، فان لم يخزن بعد فان رسالة خطأ مناسبة ترسل إلى واجهة المستخدم، بعد ذلك تقوم بجلب الخيارات التي اختارها المستخدم في صندوق الحوار (Assembler Options)، ثم جلب اسم الدليل الذي سوف يوضع فيه الملف الهدف، ثم استدعاء الملف التنفيذي(TASM32.EXE) لغرض تجميع البرنامج وتكوين الملف الهدف (OBJ.) وكذلك تكوين ملف القائمة (LST.)، وأخيراً استعراض نتائج العملية على نافذة محث (MS – DOS).

٢-٤.عملية ربط البرنامج (Linking):

في هذه الكتلة يتم التأكد من وجود الملف الهدف الذي تم تكوينه في عملية التجميع، فإن لم يوجد فان رسالة خطأ مناسبة سوف ترجع إلى واجهة المستخدم، وإلا فسيتم جلب الخيارات التي اختارها المستخدم في صندوق الحوار (LinkerOptions)، ثم جلب الملف الهدف، ثم استدعاء الملف التنفيذي (TLINK.EXE) لغرض ربط البرنامج وتكوين ملف تنفيذي للبرنامج (EXE.) وملف خريطة (MAP.)، ثم استعراض نتائج العملية على محث (MS – DOS).

2-٥ . تنفيذ البرنامج (Executing):

يتم البحث عن الملف التنفيذي الخاص بالبرنامج في النافذة النشطة، فاذا لم يوجد يتم إرسال رسالة خطأ مناسبة، وإلا فسيتم استدعاء الملف التنفيذي وإظهار نتائج البرنامج على شاشة (MS – DOS).

٢-٦. واجهة المستخدم (User Interface):

وهي اكبر واهم كتلة موجودة في البرنامج المحرر لأنها المسؤولة عن حفظ التواصل بين البرنامج المحرر والمستخدم. ولواجهة المستخدم وظائف منها :

- *الوظائف الداخلية :* وهي الوظائف التي تقوم بها الواجهة دون إشعار المستخدم، مثل تحميل الكلمات المحجوزة للغة من قاعدة البيانات إلى المصفوفة الخاصة.
- *الوظائف الخارجية :* وهي الوظائف التي تقوم بها الواجهة أما بطلب من المستخدم أو لتنبيه المستخدم عن حصول خطأ ما في البرنامج، ومن أمثلة هذه الوظائف عملية فتح وحفظ الملفات وإرجاع رسائل خطأ من بقية أجزاء البرنامج.

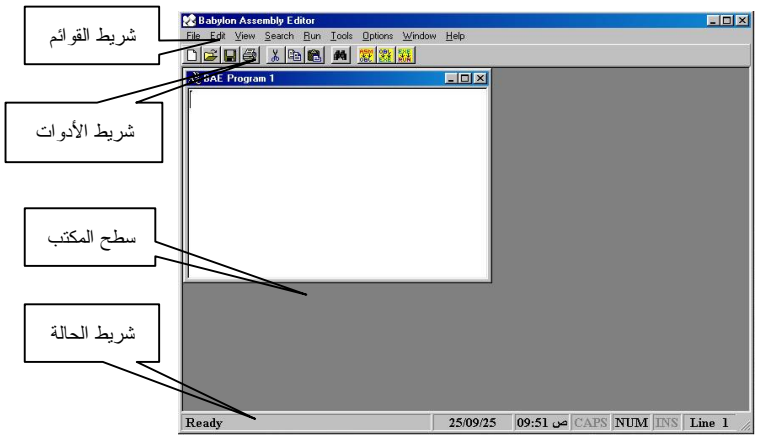
٣ . المكونات البرمجية لبيئة Babylon :

تتكون الواجهة البرمجية لبيئة Babylon من أربعة أجزاء أساسية هي شريط القوائم (Menu Bar) وشريط الأدوات (Tool Bar) وسطح المكتب (Desktop) وشريط الحالة (Status Bar) وكما في الشكل (٢).

٣-١. شريط القوائم (Menu Bar) :-

يحتوي على قوائم تمثل جميع الأوامر التي يمكن التعامل معها في هذه البيئة، ويمكن استخدامها بطريقة تشبه استخدام القوائم في نظام Windows ، وتتكون من:

اولاً : قائمة الملف (File Menu) : تحتوي قائمة الملف على أوامر لإنشاء، وفتح، وإغلاق، وحفظ، وطباعة الملفات، واخيراً الخروج الى نظام Windows ولاختيار قائمة الملف تظهر قائمة شبيهة بالشكل (٣):



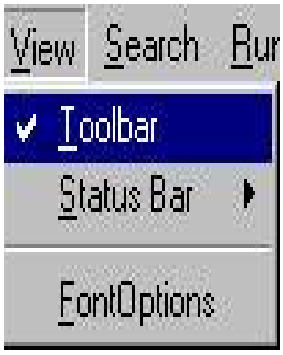
الشكل (٢) : مكونات البرنامج

الشكل(٣)قائمة الملف

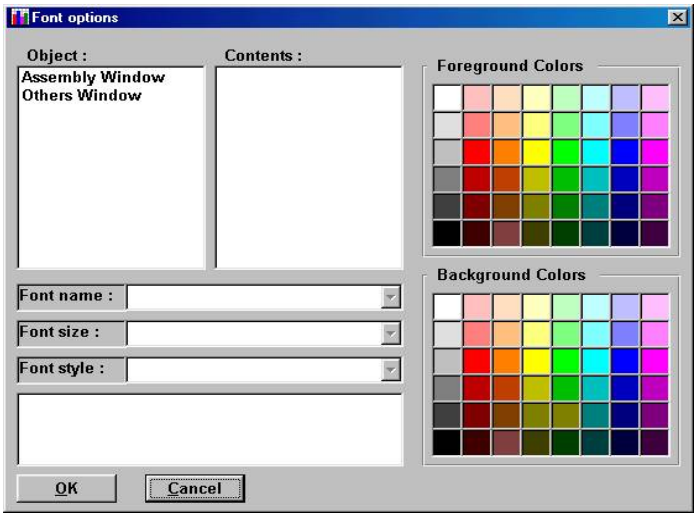
ثانياً: قائمة التحرير (Edit Menu) : تمكننا أوامر قائمة التحرير الظاهرة في الشكل (٥) من القيام بعمليات على النصوص، حيث تعمل معظم أوامر هذه القائمة بعد تعليم النصوص (Mark) أو اختيارها (Select)، واهم هذه العمليات هي نقل النصوص بين الملفات، حيث يقوم الخيار نسخ لنسخ هذه النصوص إلى الحافظة (Clipboard) كموقع وسيط بين هذه الملفات. ثم تتم عملية اللصق التي تظهر ما موجود في الحافظة .

ثالثاً: قائمة العرض (View menu) : تتحكم هذه القائمة الموضحة في الشكل (٦) بكل ما يمكن التحكم بإظهاره داخل البيئة ، من خلال أوامر هي **Tool Bar**: الذي يستخدم لإظهار وإخفاء مستطيل الأدوات.و **Status Bar** : الذي يستخدم بالتحكم بإظهار وإخفاء مستطيل الحالة أو أحد محتوياته. وأخيراً **Font Options** : الذي يستخدم لاستدعاء صندوق الحوار الخاص بخيارات الخط كما في الشكل (٧) :

الشكل (٥) قائمة التحرير



شكل (٦) قائمة العرض



شكل (٧) خيارات الخط

رابعاً: قائمة البحث (Search menu): تتضمن قائمة البحث الظاهرة في الشكل (٨) على الأمر الأول **Find** الذي يبحث عن نص معين، وتحدد خيارات هذا الأمر من خلال الشكل (٩) الذي يظهر صندوق حوار يحوي على بعض الخيارات الأخرى. أما الأمر الأخر فهو **Find Next** الذي يكرر آخر عملية بحث. وأخيراً الأمر **Replace** الذي يستخدم للبحث عن نص محدد واستبداله بآخر.

الشكل (٨) قائمة البحث

شكل (٩) صندوق الحوار البحث عن نصوص

خامساً: قائمة التنفيذ (Run menu) : يمكننا من خلال أوامر هذه القائمة الظاهرة في الشكل (١٠) ترجمة البرامج وربطها وتنفيذها، وهذه الاوامر هي: **Assemble** الذي يقوم بتجميع البرنامج (Assembling) الموجود في النافذة النشطة. والامر **Link** الذي يقوم بعملية الربط (Linking) للبرنامج الموجود في النافذة النشطة. وأخيراً الأمر **Run** الذي ينفذ البرنامج الموجود ضمن النافذة النشطة عن طريق استدعاء الملف التنفيذي الذي تكون نتيجة عملية الربط.

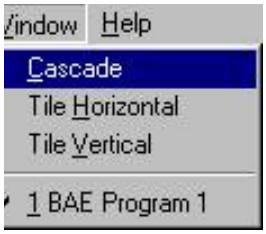
سادساً: قائمة الأدوات (Tools menu) : تحتوي هذه القائمة على بعض البرامج والأدوات التي تفيد المبرمج أثناء العمل مع لغة التجميع، مثل الامر: **ASCII Table** الذي يستدعي نافذة خاصة تحتوي على كل المحارف (Characters) الموجودة ضمن شفرة (ASCII) . والامر **Calculator** الذي يقوم باستدعاء برنامج الحاسبة الذي يأتي مع نظام (Windows)، وهذه الحاسبة مفيدة جداً إذا ما استخدمت كحاسبة علمية (Scientific)، إذ تحتوي على كل ما يحتاجه المبرمج من تحويلات الأنظمة العددية والعمليات الرياضية عليها.

سابعاً: قائمة الخيارات (Options menu): تعدل هذه القائمة الظاهرة في الشكل (١١) ببيئة العمل من حيث الخيارات للمجمع والرابط ومواقع خزن العمل، وتحفظ هذه التغييرات. وللوصول إلى هذه القائمة نضغط Alt+O .

ثامناً: قائمة النافذة (Window menu) : تزودنا القائمة الظاهرة في الشكل(١٢) بمجموعة أوامر تستخدم للتحكم بطريقة ترتيب وإظهار النوافذ على الشاشة وللوصول إلى هذه القائمة نضغط Alt+W.

تاسعاً: قائمة المساعدة (Help menu): نستطيع من خلال هذه القائمة الواضحة في الشكل (١٣) الوصول إلى فهرس بالمحتويات، أو الأسماء، أو حتى معظم محتويات جداول للمساعدة، ويمكننا انتقاء قائمة المساعدة تلك بالضغط على Alt+ H.

الشكل (١٣) قائمة المساعدة



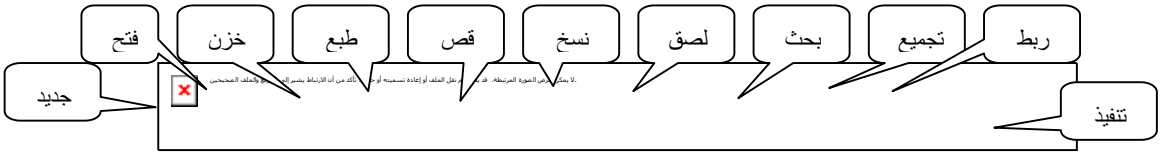
الشكل (١٢) قائمة

الشكل (١١) قائمة

شكل (١٠) قائمة التنفيذ

٣-٢. شريط الأدوات (Tool Bar) :

يحتوي شريط الأدوات الموضح في الشكل (١٤) على بعض الأوامر المهمة والتي يتكرر استخدامها بكثرة أثناء العمل ويمكن الوصول إليها بسهولة بالضغط عليها باستخدام الفارة.



الشكل(١٤) شريط الأدوات

٣-٣. سطح المكتب (Desktop) :

يحتل سطح المكتب غالبية مساحة شاشة عرض البيئة البرمجية، حيث يحتوي على نوافذ التحرير و كل صناديق الحوار ورسائل خطأ التي تظهر على سطح المكتب وكما موضح في الشكل (٢).

٣-٤. شريط الحالة (Status Bar) :

يعرض في هذا الشريط (الذي يقع في الجانب السفلي من واجهة البيئة البرمجية) معلومات وملاحظات مهمة ومفيدة للمستخدم تتغير حسب الحالة التي تكون فيها البيئة في ذلك الوقت كتحديد الوقت والتاريخ و حالة المفاتيح Caps و NUM و INS وكما هو موضح في الشكل (٢).

4- دراسة حالة عملية

في هذه الفقرة سنأخذ مثالاً عملياً عن كيفية تعامل المحرر مع الملفات المصدرية عند التحرير وعند التنفيذ. وسنبدأ بتكوين ملف جديد عن طريق قائمة الملف (File menu) الأمر (New) فتظهر نافذة تحرير على سطح المكتب الخاص بالمحرر تحمل أسم (1 BAE Program)، والآن البرنامج جاهز للكتابة والتحرير. وكما في الشكل (٢):

نقوم بكتابة البرنامج التالي (كحالة للدراسة)، وهو برنامج بلغة التجميع في نافذة التحرير التي فتحناها قبل قليل:

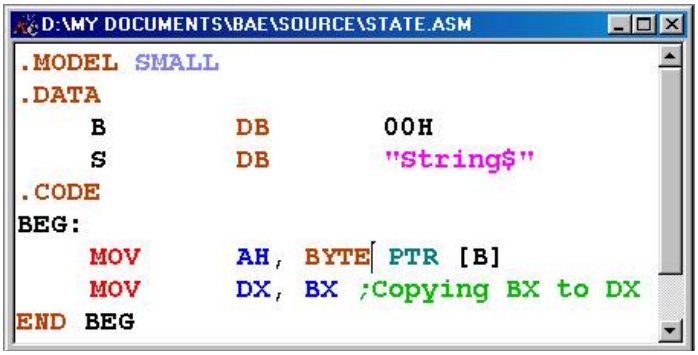
```
.MODEL SMALL
.DATA
        B       DB      00H
        S       DB      "String$"
.CODE
  BEG:
```

```
        MOV   AH, BYTE PTR [B]
        MOV DX, BX  ; copying BX to DX
    END BEG
```

نلاحظ أثناء كتابة هذا البرنامج أننا كلما انهينا كتابة أحد اسطر البرنامج فان اللون الأسود للكتابة سوف يتحول إلى ألوان أخرى تعتمد على نوعية الكلمة المكتوبة فيبدو البرنامج في داخل نافذة التحرير مشابه للشكل (15).

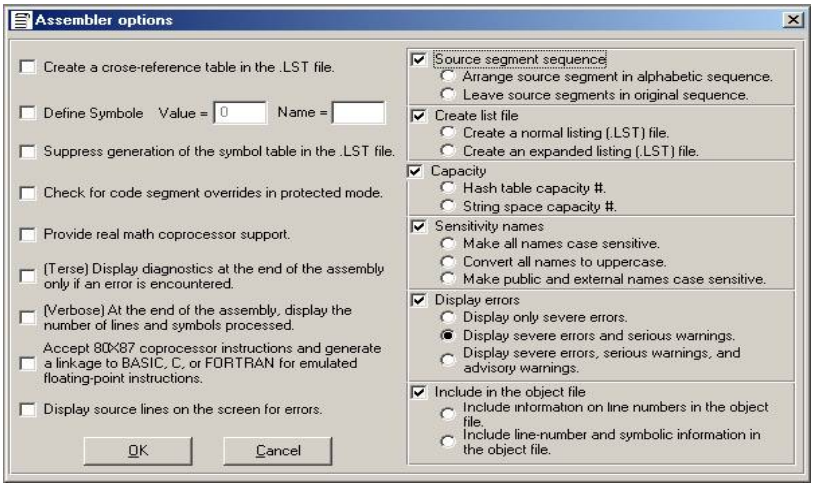
بعد إكمال كتابة البرنامج نقوم بخزن البرنامج على القرص وباسم معين وليكن (State.asm)، وذلك لكي نقوم بتنفيذه، لان البرنامج المصدري لا ينفذ مباشرة من النافذة وإنما عن طريق تحميل البرنامج من الملف، وإذا حصل وبدأنا تنفيذ البرنامج قبل خزنه فان رسالة خطأ شبيهة بالشكل (١٦) سوف تظهر لتنبيه المستخدم بان البرنامج لم يتم خزنه بعد.

الشكل (١٦) رسالة خطأ تظهر عند التنفيذ قبل



الشكل(١٥): البرنامج بشكله

وبعد خزن البرنامج نقوم بأول مرحلة من مراحل التنفيذ وهي مرحلة التجميع (Assembling)، لتحويل ملف البرنامج من ملف مصدري (Source File) إلى ملف هدف (Object File) ويكون امتداده (OBJ.). تتم هذه العملية عن طريق استدعاء ملف تنفيذي خارجي خاص بهذه العملية وهو الملف (TASM32.EXE). وكذلك يمكن للمبرمج أن يغير في خيارات عملية التجميع عن طريق صندوق حوار خيارات التجميع الذي يمكن إظهاره بالذهاب إلى قائمة الخيارات (Options menu)، ثم اختيار الأمر (Assembler) منها، فيظهر صندوق حوار يحتوي على كل الخيارات الخاصة بالمجمع (Turbo Assembly 5.0) من شركة (Borland) في حالة التجميع، والتي يمكن من خلالها تغيير الاختيارات حسب الطلب وكما هو واضح في الشكل (١٧) على سبيل المثال .

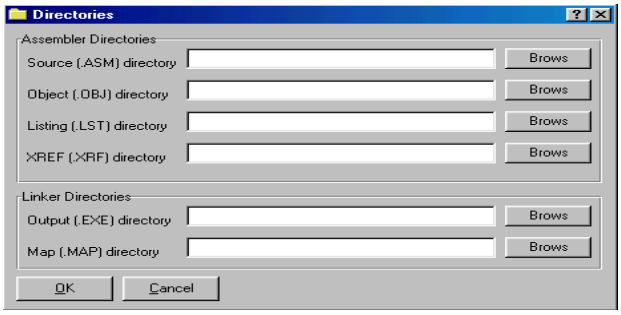


الشكل(١٧): صندوق حوار خيارات المجمّع

ويوضع الملف الهدف في دليل خاص يتم تحديده عن طريق صندوق حوار خاص بالأدلة التي يتعامل معها البرنامج وكما مبين في الشكل (١٨)، ويمكن الوصول إلى صندوق الحوار هذا عن طريق قائمة الخيارات، ثم الأمر (Directories).

ويجب أن يكون مكان الدليل بداخل الدليل الرئيسي للبرنامج وذلك لأسباب تنظيمية لتخزين الملفات الناتجة، وآلا فان رسالة خطأ شبيهة بالشكل (١٩) تظهر لتنبيه المبرمج أن الدليل خارج نطاق الدليل الرئيسي.

الشكل (١٩) رسالة خطأ للأدلة الخارجة عن نطاق البرنامج

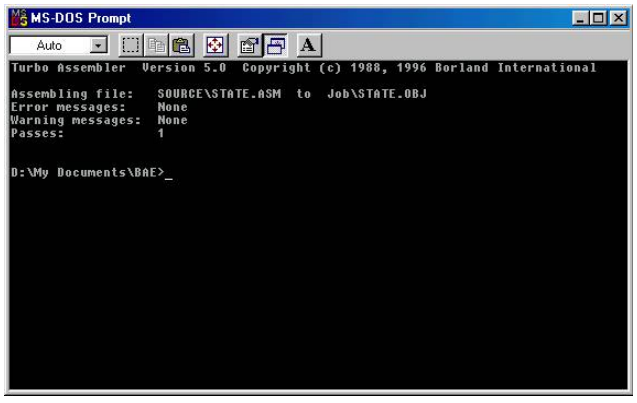


الشكل (١٨) صندوق حوار أدلة البرنامج

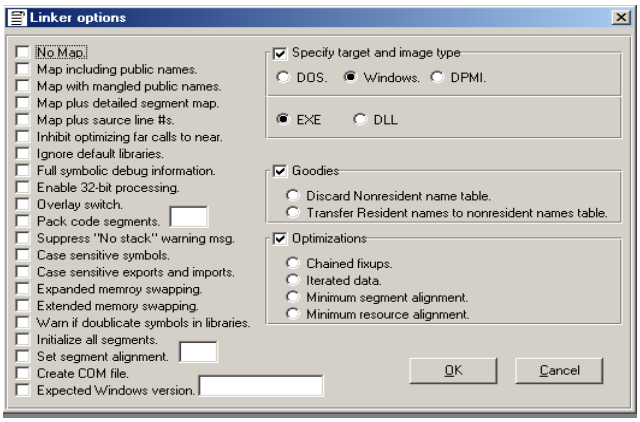
وبعد ذلك يتم تجميع البرنامج (TASM32.EXE) فتظهر نافذة سوداء خاصة بالمحث (DOS) وكما في الشكل (٢٠) تبين نتائج عملية التجميع.

وبعد إغلاق هذه النافذة، نبدأ بالمرحلة الثانية من مراحل التنفيذ وهي مرحلة الربط (Linking)، وهي عملية تحويل الملف الهدف (Object File) إلى ملف تنفيذي (Execution File)،

حيث يمكن للمبرمج أن يغير في خيارات عملية الربط عن طريق صندوق حوار خيارات الربط الذي يمكن إظهاره بالذهاب إلى قائمة الخيارات (Options menu)، ثم اختيار الأمر (Linker) منها، فيظهر صندوق حوار شبيه بالشكل (٢١)، يمكن من خلالها تغيير الخيارات المطلوبة.



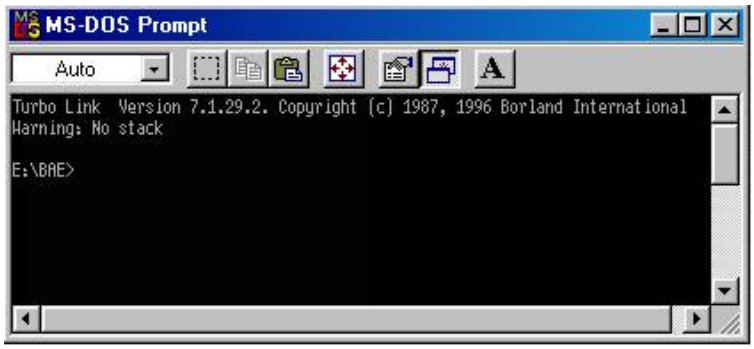
الشكل (٢٠) نتائج عملية التجميع



الشكل(٢١): صندوق حوار خيارات الرابط

بعد اخيار الخيارات المطلوبة واغلاق صندوق الحوار يتم تحويل البرناج الهدف الى البرنامج التنفيذي عن طريق استدعاء الملف الخارجي (Tlink.exe) والتابع لشركة (Borland) لتظهر نافذة مشابه للشكل (٢٢)

بعد ذلك يكون البرنامج القابل للتنفيذ، وبالذهاب إلى قائمة التنفيذ واختيار الأمر (Execute)، فتظهر نتائج تنفيذ البرنامج في نافذة المحث DOS .



الشكل (٢٢) نتائج عملية الربط

٥-النتائج

١- أن البيئة البرمجية التي تم تصميمها وبنائها هي بيئة صديقة للمبرمج وسهلة الاستخدام تبدو وكأنها جزء من نظام الـ Windows .

٢- تم تصميم وبناء محرر كامل خاص لتحرير برامج لغة Assembly .

٣- سرعة المعالجة للبيئة الجديدة تعتمد على سرعة الحاسب وحجم الملف.

٤- إضافة تقنية الملفات المساعدة المباشرة (On Line Help Files) إلى البيئة لمساعدة المبرمج في الحصول على معلومات آنية في لغة Assembly .

٦-المقترحات والأعمال المستقبلية

١-بناء مجمّع (Assembler) و رابط (Linker) داخليين (Built – in) وليس باستدعاء ملفات تنفيذية خارجية.

٢-بناء منقّح داخلي (Built – in Debugger) يقوم بمراقبة تنفيذ البرنامج، وهذا يجعل البيئة البرمجة أكثر قوة.

٣- إيجاد طريقة تجعل من عملية المعالجة اكثر سرعة وخاصة عند تحميل الملفات الكبيرة جداً ، حيث ان عملية البحث عن كلمة ثم إرجاع اللون المناسب لها هي عملية طويلة نسبياً.

٧- المصادر

ورنر فيبل، **تربو باسكال ٧-الدليل الكامل-**، شعاع للنشر والعلوم، الطبعة الأولى، ١٩٩٥.

انيس محمد توفيق، **فيجول بيسك ٥**، دار الراتب الجامعية-لبنان، ١٩٩٨.

Microsoft MS-DOS User's Guide version 3.3, Microsoft Corporation, 1987.

Microsoft MSDN Library, Microsoft Corporation, January, 2000.

Peter Abel, **IBM PC ASSEMBLY LANGUAGE AND PROGRAMMING**, Prentice-Hall International Inc., Fourth Edition, 1998.

Ellis Horowitz, **Computer Algorithm / C++**, Prentice-Hall International Inc., 1997.

Eric Winemiller, Jaso T. Roff, Bill Heyman and Ryan Groom, **Visual Basic 6 Databases HOW-TO**, Macmillan Computer Publishing, 1999.