الجلسة الأولى

مدخل إلى المترجمات (مراحل الترجمة، والمتطلبات البرمجية)

Introduction to Compilers (Compiler Stages and Software Requirement)

الهدف من الجلسة

- التعريف بمفهوم المترجمات العام وكيفية بناء المترجم.
- التعريف بمراحل التحليل اللفظي والقواعدي والمعنوي وتوليد الشيفرة.

مستلزمات الجلسة

- حاسب بمواصفات دنيا RAM: 1 GB, CPU: 1.6 GHz, Windows 10 OS 64 bit
- Turbo c++/ DevC++
- LEX & BISON tools

خطوات العمل

1. ماهو المترجم؟
2. ماهي مراحل الترجمة؟
2. 1. المرحلة الأولى– التحليل اللفظي
2. 2. مرحلة التحليل القواعدي
2. 3. مرحلة التحليل المعنوي
2. 4. مرحلة توليد الكود
3. ماهي المتطلبات الأساسية لتوليد المترجمات؟
3. 1. بنية ملف وصف الماسح Scanner
3. 2. بنية ملف وصف المعرب Parser
4. الخطوات العملية لبناء مترجم بسيط

الخلاصة والنتائج:

يفترض عند نهاية الجلسة:

تمكن الطالب من فهم كيفية مكونات المترجم وآلية عمل المترجم بشكل عام.

1-1 ماهو المترجم؟

بفرض لدينا برنامج مكتوب بلغة عالية المستوى (مثل الفيجوال بيسك والسي) وسيتم تنفيذه من قبل الحاسوب، هذا يتطلب أن يكون لدينا مرحلة وسيطة تجعل الحاسوب قادرا على فهم هذه اللغة نظرا لأن الحاسوب يعمل على لغة الآلة (الأصفار والواحدات)، وبالتالي تكون عملية الترجمة هي تلك العملية التي يتم بها تحويل الكود من لغة المصدر إلى لغة الآلة والمترجم هو برنامج مكتوب بلغة ما يقوم بترجمة كود من لغة مصدر إلى لغة الآلة.

يمكن توضيح هذه العملية كما في المخطط التالي [1]:

الشكل(1-1) مخطط مراحل الترجمة

2-1 ماهي مراحل الترجمة؟

تقسم عملية الترجمة لعدة مراحل كالآتي:

1. التحليل اللفظي
2. التحليل القواعدي
3. تحليل المعاني
4. توليد الشيفرة الوسيطية
5. توليد الشيفرة النهائية

1-2-1 المرحلة الأولى- التحليل اللفظي Lexical Analysis:

هي المرحلة الأولى من التحليل يقوم بها المترجم باستخراج المفردات والتي نسميها Tokens انطلاقا من سلسلة الدخل اعتماداً على قوالب محددة مسبقاً تم تعريف المترجم عليها.

مثلاً: قالب أرقام، قالب لأسماء المتغيرات، قالب لكلمة محجوزة الخ.... وتسمى هذه القوالب بالتعابير المنتظمة Regular Expressions.

وعلى سبيل المثال ليكن لدينا الكود البرمجي التالي:

```
For     i:= 1 to vmax  do  a :=  a+i ;
```

جدول (1-1) كيفية استخراج جدول الـ Tokens

نوع المفردة Tokens	ماهيتها
كلمة محجوزة	For
معرف	I
إسناد	:=
عدد صحيح	1
كلمة محجوزة	To
معرف	Vmax
كلمة محجوزة	do
معرف	a
إسناد	:=
عملية رياضية	+
معرف	i
فاصلة نهاية التعليمة	;

يلي ذلك مرحلة بناء جدول الرموز (Symbol Table)، وهي عبارة عن قائمة من تركيبات تحمل خصائص المفردات Tokens التي تم تحديدها في الخطوة السابقة:

جدول (1-2) جدول الرموز

رقم الرمز	Token	نوع Token	نوع variable
1	For	كلمة محجوزة	–
2	To	كلمة محجوزة	–
2	Do	كلمة محجوزة	–
3	;	فاصل	–

4	I	معرف	Integer
.	.	.	.
.	.	.	.

إذاً تمثل هذه المرحلة مسحاً Scanning لتسلسل الدخل لتحديد المفردات Tokens الداخلة في تركيب العبارة، ولا يهتم المحلل اللفظي بترتيب الـ Tokens وبالتالي السطر التالي صحيح تماما بالنسبة له:

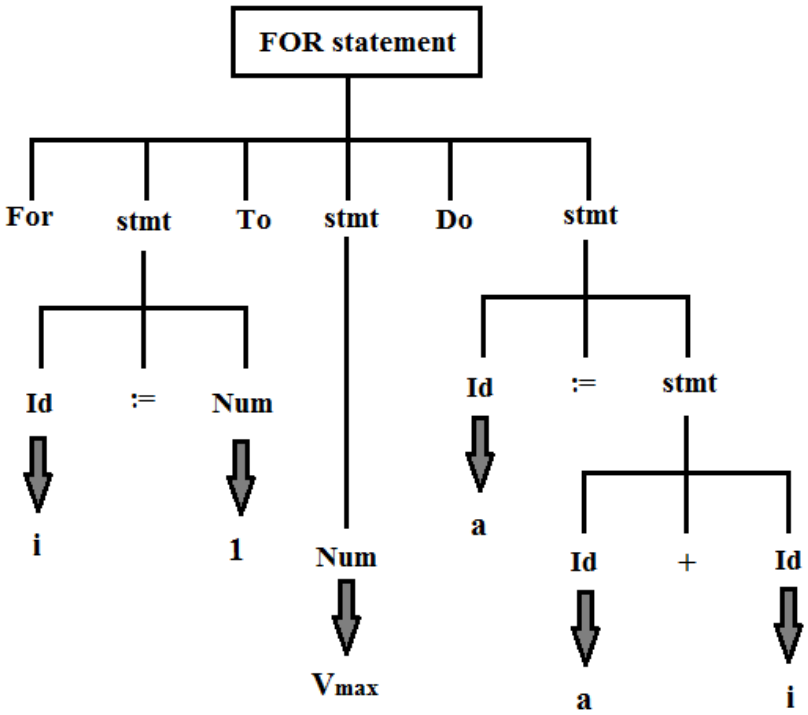
```
For  for  for  i  :=  :=  10  do  for  a  a;
```

تعد هذه العبارة صحيحة لفظياً Lexically correct لكنها خاطئة قواعدياً، وهنا تأتي مرحلة التحليل القواعدي لتحديد هذه الأخطاء.

2-2-1 مرحلة التحليل القواعدي Syntax Analysis:

يتم خلال هذه المرحلة التحقق من النحو الخاص باللغة ويكون هذا وفقا لمجموعة من القواعد والتي تمثل الضابط الذي يحكم صحة تسلسل المفردات Tokens التي تم مسحها من قبل المحلل اللفظي.

يقوم المحلل القواعدي ببناء الشجرة باستخدام مجموعة Tokens التي تم توفيرها بالجدول السابق، مع الأخذ بعين الاعتبار قواعد اللغة التي ستحدد فيما إذا كانت العبارة الممسوحة صحيحة قواعدياً أو لا.



الشكل (1-2) شجرة الإعراب الخاصة بعبارة حلقة for السابقة

3-2-1 مرحلة تحليل المعاني Semantic Analysis:

قد تكون العبارة صحيحة لفظياً وقواعدياً، لكنها ليست ذات معنى.

تعتمد هذه المرحلة بشكل أساس على جدول الرموز المنشأ سابقاً والذي يتضمن أسماء الرموز التي صادفها الماسح خلال مسحه لسلسلة الدخل.

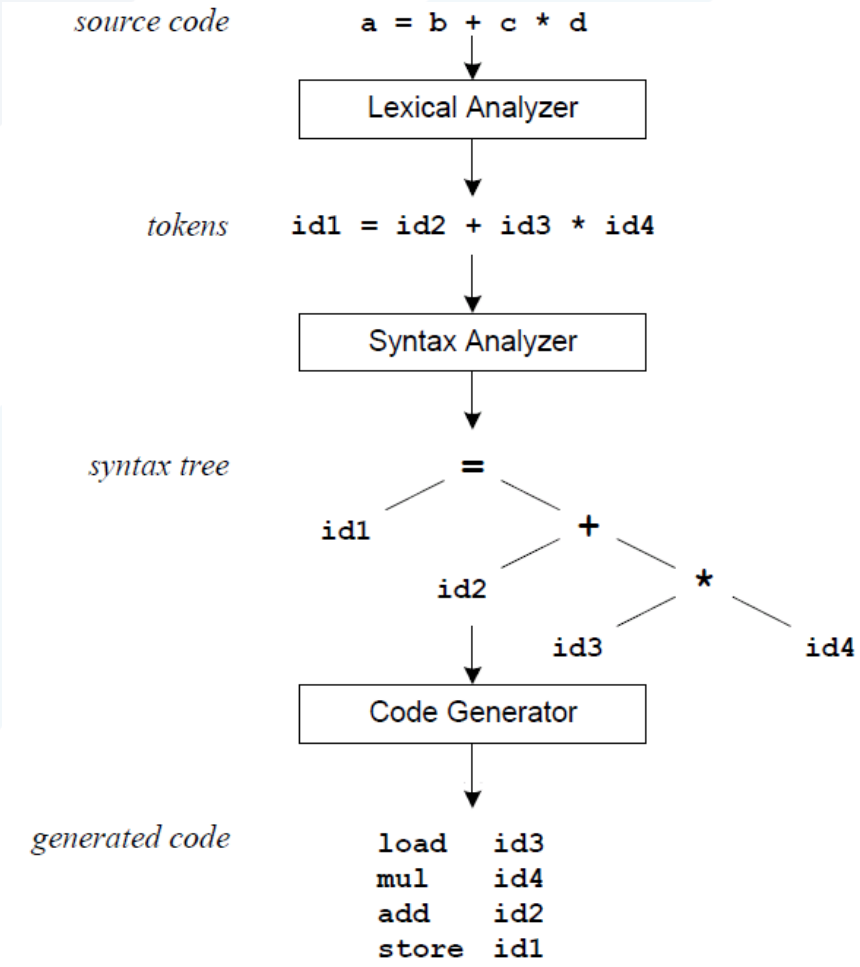
هناك العديد من الشروط التي يتم التأكد منها في هذه المرحلة مثل:

- التأكد من تطابق أنواع المتغيرات مثل إسناد رقم حقيقي real لمتغير من النوع الصحيح integer.
- التأكد من تطابق نوع المتغير مع قيمته.
- التأكد من عدم تكرار اسم متغير معرفاً مسبقاً
- التأكد من عدم إسناد قيمة لمتغير غير مصرح عنه سابقاً

4-2-1 مرحلة توليد الكود Code generation:

وهنا نقوم بعملية إنتاج كود بلغة الآلة أو لغة التجميع.

مثال بسيط: ماهي مراحل الترجمة اللازمة لترجمة العبارة التالية: A=B+C*D



3-1 ماهي الأدوات الأساسية لبناء المترجمات؟

يوجد عدد من الأدوات البرمجية التي يمكن استعمالها لبناء المترجمات ولكن يوجد هناك أداتين أساسيتين هما الأكثر شهرةً لذلك هما:

- LEX
- BISON

1-3-1 الأداة البرمجية (LEX):

وهي أداة تقوم بتوليد محلل مفردات أو ما يتم تسميته عادة بـ Scanner مكتوب بلغة c حيث يتم استخدام قوالب جاهزة لمطابقة السلاسل المحرفية الموجودة في الكود المصدر وتحويلها إلى tokens

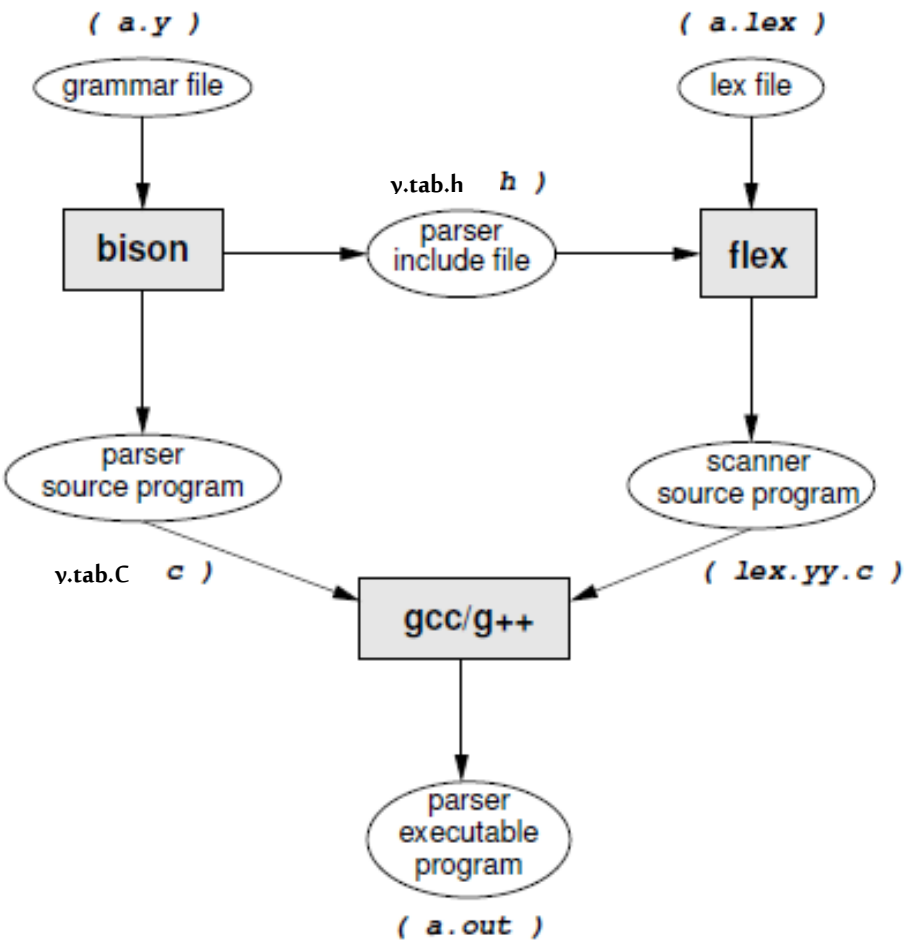
مثلاً عندما يصادف الماسح (Scanner) المتغير X في سلسلة الدخل فإنه يرسل المفردة (token) التي تمثل هذا المتغير إلى المحلل القواعدي مثلاً <id>، كما يرسل له أيضا اسم المتغير ويقوم بإدخاله إلى جدول الرموز وتعيين خصائصه كنوعه وقيمته وغيرها من الخصائص.

2-3-1 الأداة البرمجية (BISON or YACC):

تولد هذه الأداة شيفرة بلغة C لمحلل قواعدي ويعرف أيضا بالمعرب (parser). يستعمل المعرب قواعد اللغة لتحليل الـ Token القادمة من الـ Scanner ويبني منها شجرة الإعراب الموافقة لتسلسل الـ Token.

في حالة عدم مطابقة العبارة للنحو، يصدر الـ BISON رسالة خطأ اعتماداً على تابع الخطأ المزود به (سيتم استعراضه لاحقاً عند شرح بنية ملف parser).

يوضح الشكل (1-3) مراحل ترجمة ملف مصدري باستخدام أداتي (LEX,BISON).



الشكل (1-3) مراحل بناء مترجم باستخدام أداتي LEX وَ BISON

وحسب الشكل (1-3)

a.y يمثل وصف المعرب Parser description file

الملف a.L فيمثل ملف وصف الماسح Scanner description file

الملف y.tab.h فيولده برنامج bison ويحتوي تعريفات المفردات Tokens مثل:

```
#define     IDENT     102
```

بعد ذلك يولد لنا كل من LEX و Bison الملفات:

lex.yy.c وهو محلل المفردات scanner أو المحلل المعجمي (كما سميناه في النظري).

الملف y.tab.c وهو المحلل القواعدي (القواعدي) parser.

وباستعمال أحد مترجمات لغة الـ C يتم الحصول على البرنامج النهائي ذو الامتداد exe. وهو المترجم الجديد وهنا نحتاج لمترجم لغة C مثلاً GCC أو G++ لترجمة ملف المعرب فقط y.tab.c.

إذاً لا بد من معرفة محتوى ملفي وصف الماسح Scanner ووصف المعرب parser وكيف يمكن كتابتهما (الجزء المطلوب من الطالب أن يتعلم كيفية كتابته).

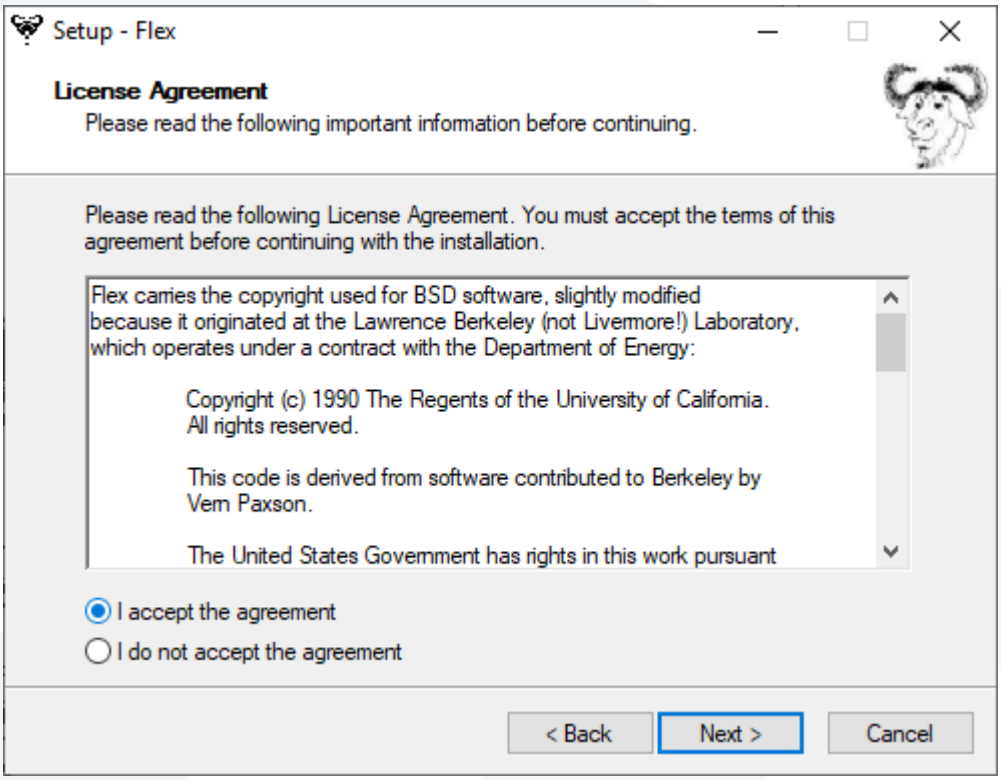
إذا سنحتاج في الجانب العملي:

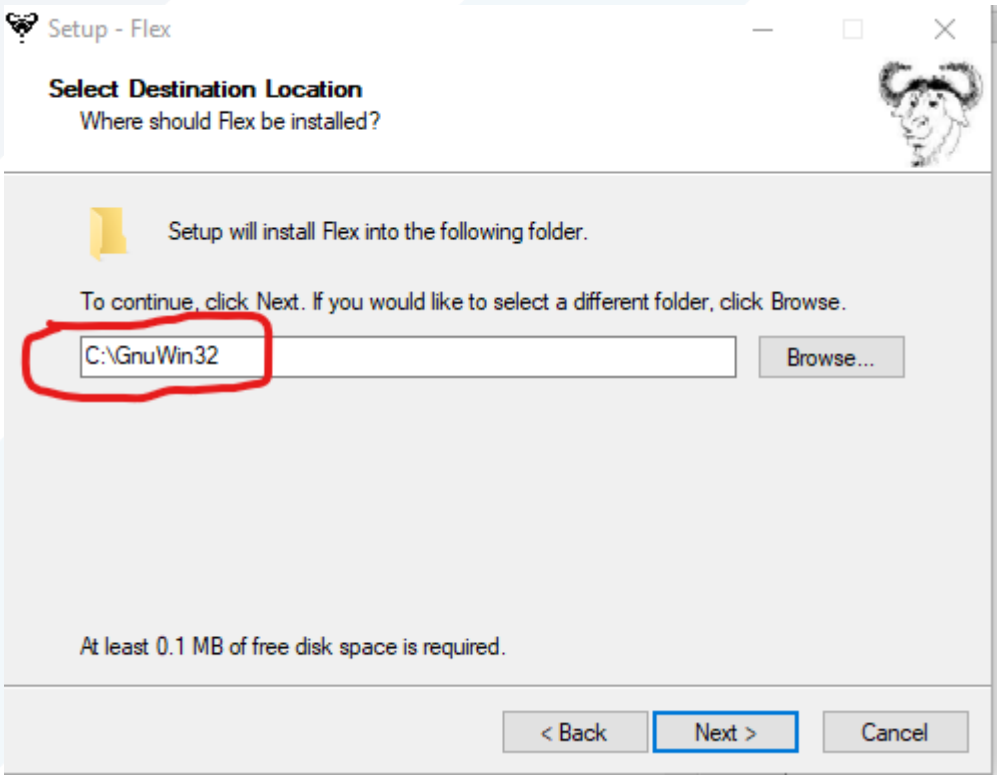
برمجيتي (YACC) Lex, BISON.

مترجم لغة C

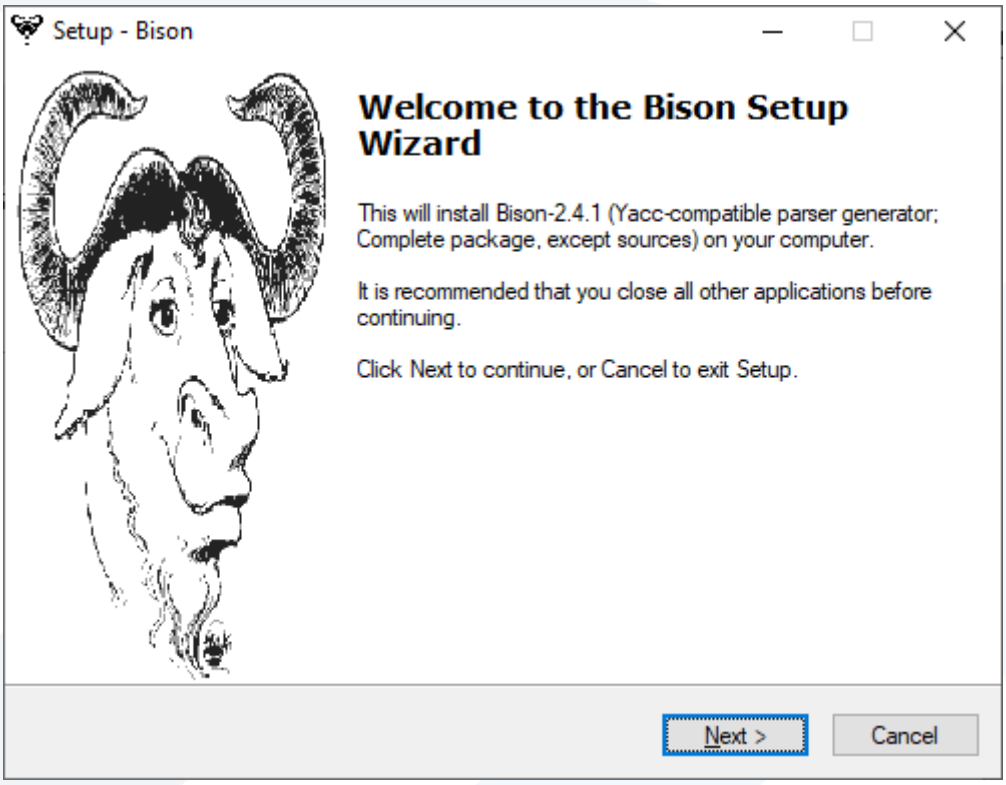
يتم تنزيل هذه البرمجيات في الجانب العملي:

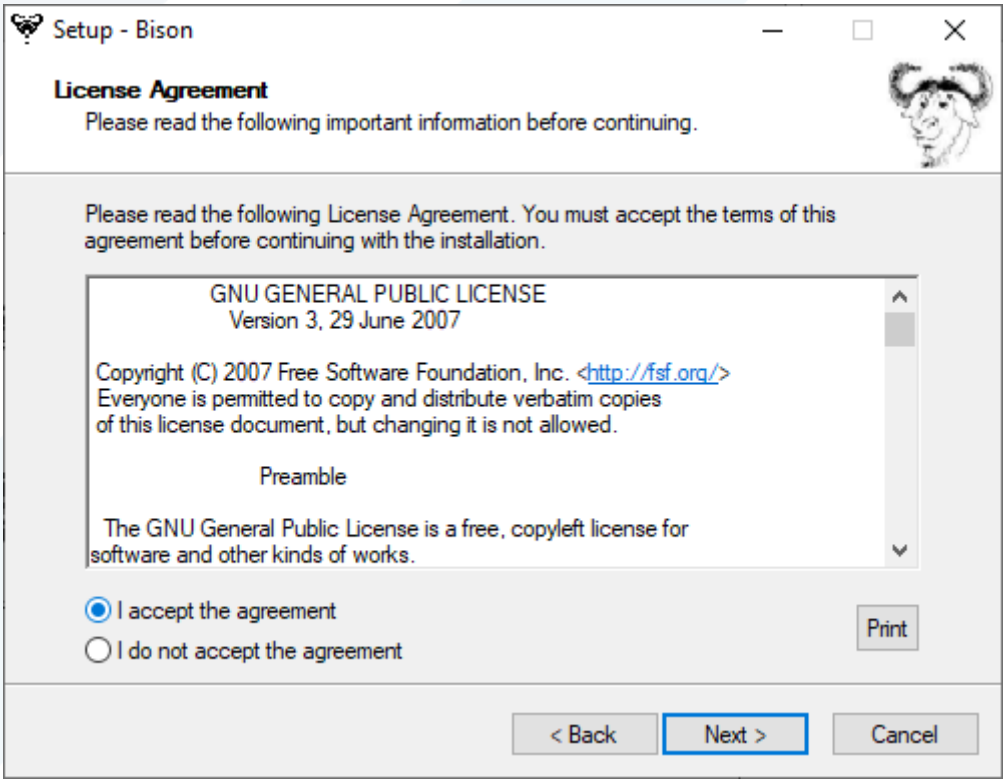
أولاً) برمجية LEX من خلال تنصيب البرنامج flex-2.5.4a-1.exe

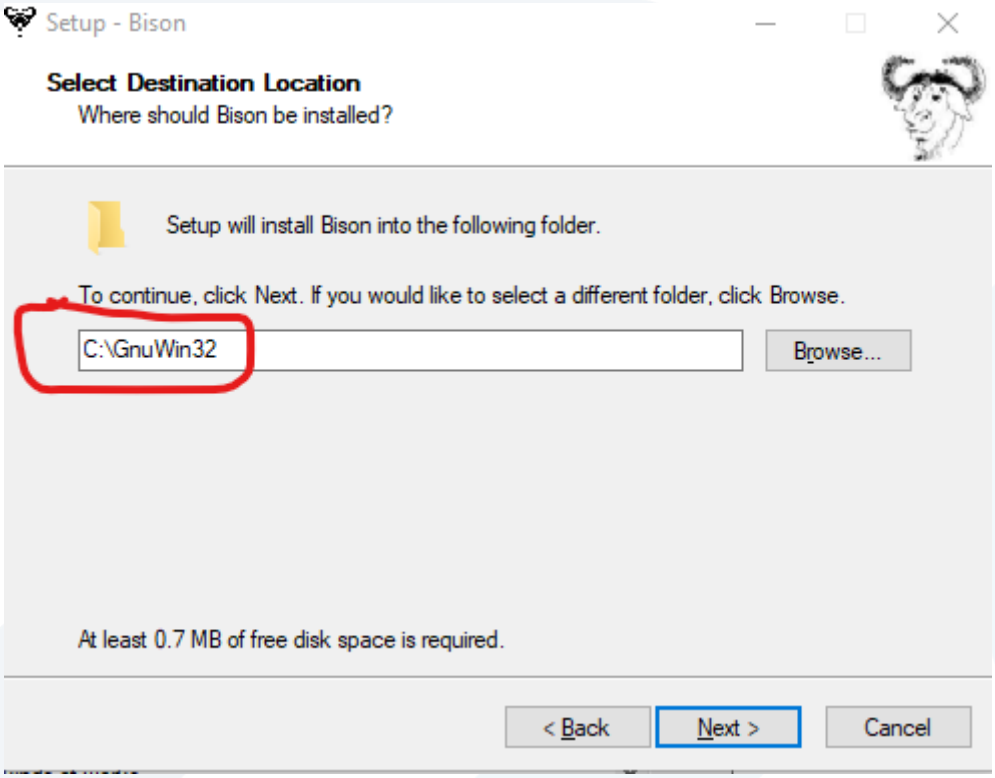




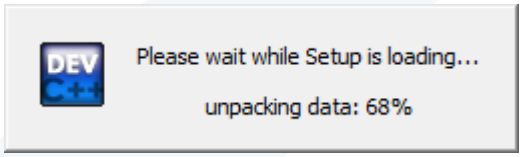
ثانياً) برمجية BISON من خلال تنصيب البرنامج bison-2.4.1-setup.exe

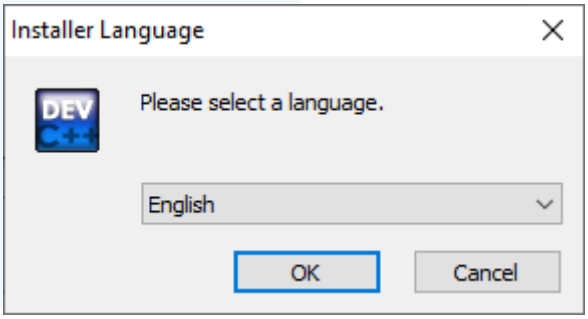

Setup - Bison
Welcome to the Bison Setup Wizard
This will install Bison-2.4.1 (Yacc-compatible parser generator; Complete package, except sources) on your computer.
It is recommended that you close all other applications before continuing.
Click Next to continue, or Cancel to exit Setup.
Next >
Cancel

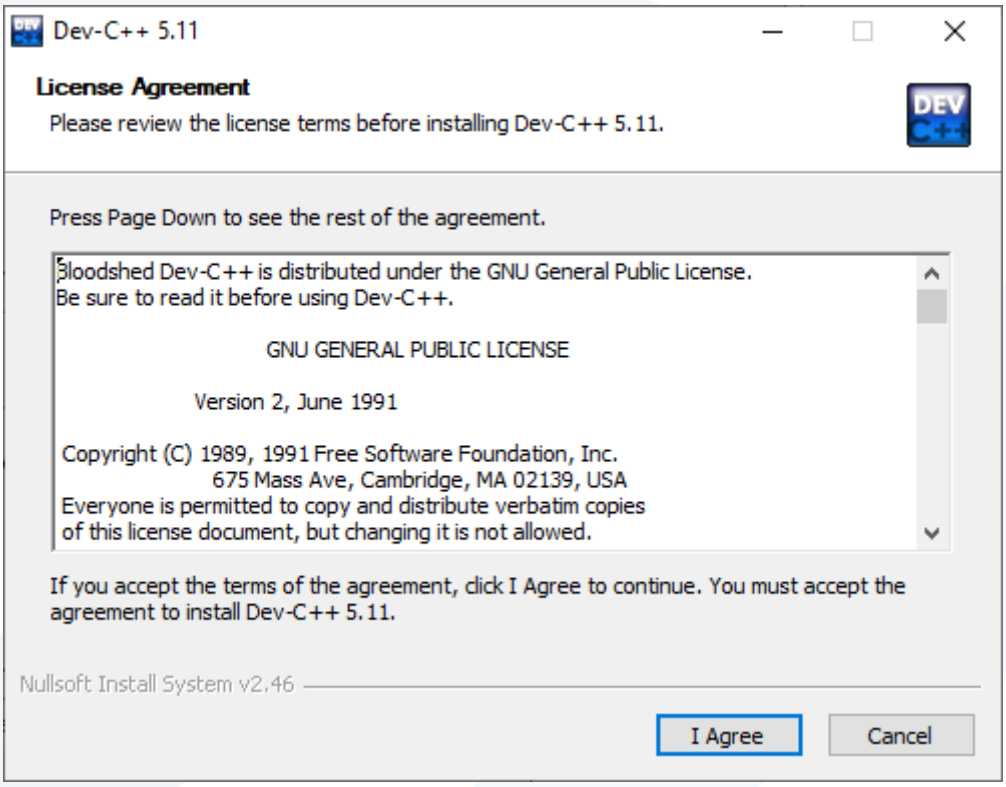

Setup - Bison
License Agreement
Please read the following important information before continuing.
Please read the following License Agreement. You must accept the terms of this agreement before continuing with the installation.
GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007
Copyright (C) 2007 Free Software Foundation, Inc. <http://fsf.org/>
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.
Preamble
The GNU General Public License is a free, copyleft license for
software and other kinds of works.
I accept the agreement
I do not accept the agreement
Print
< Back
Next >
Cancel

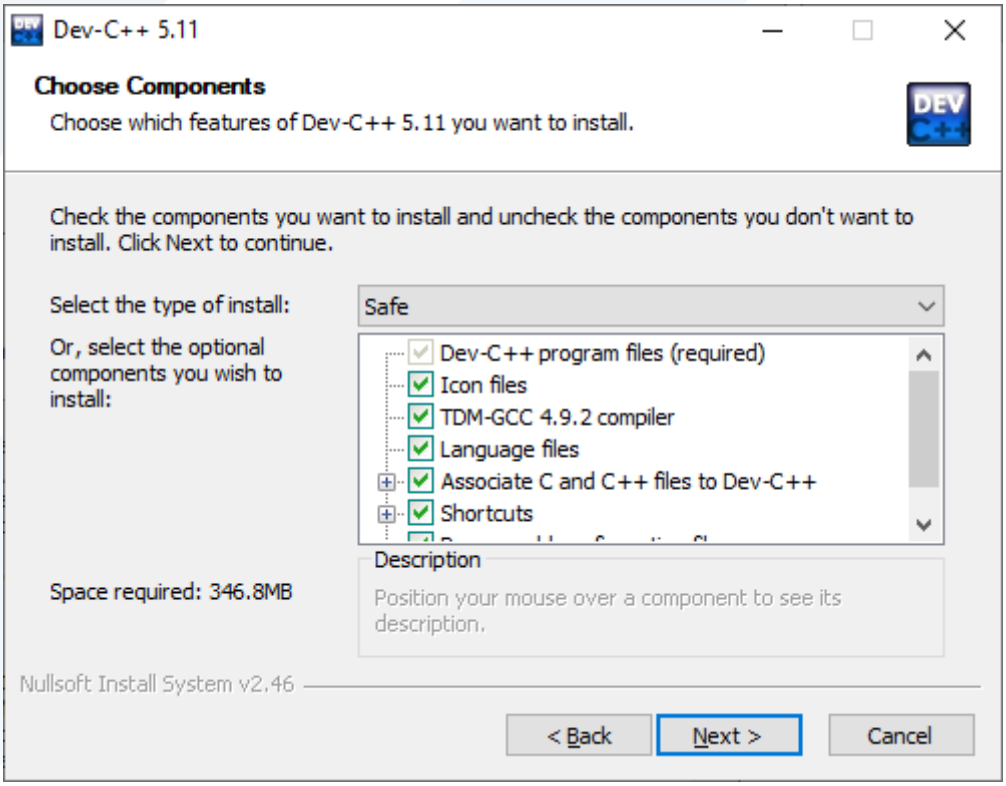


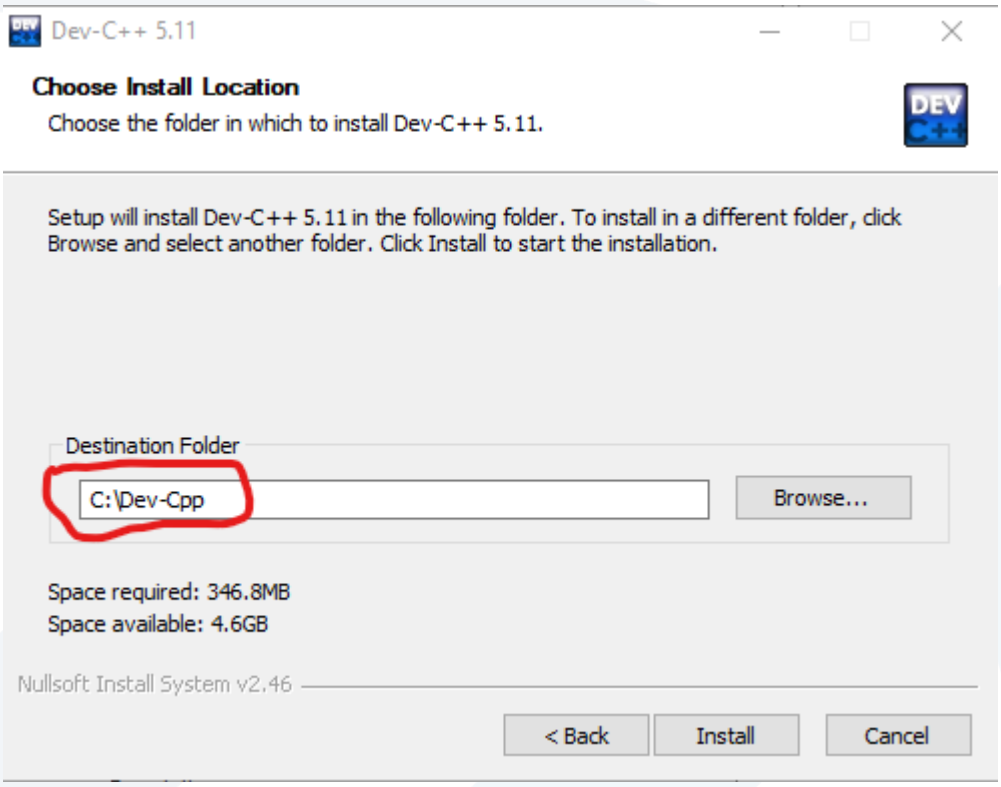
ثالثاً) مترجم لغة C من خلال تنصيب البرنامج Setup.exe TDM-GCC 4.9.2 Dev-Cpp 5.11.






Dev-C++ 5.11
License Agreement
Please review the license terms before installing Dev-C++ 5.11.
Press Page Down to see the rest of the agreement.
Bloodshed Dev-C++ is distributed under the GNU General Public License.
Be sure to read it before using Dev-C++.
GNU GENERAL PUBLIC LICENSE
Version 2, June 1991
Copyright (C) 1989, 1991 Free Software Foundation, Inc.
675 Mass Ave, Cambridge, MA 02139, USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.
If you accept the terms of the agreement, click I Agree to continue. You must accept the agreement to install Dev-C++ 5.11.
Nullsoft Install System v2.46
I Agree
Cancel


Dev-C++ 5.11
Choose Components
Choose which features of Dev-C++ 5.11 you want to install.
Check the components you want to install and uncheck the components you don't want to install. Click Next to continue.
Select the type of install:
Safe
Or, select the optional components you wish to install:
Dev-C++ program files (required)
Icon files
TDM-GCC 4.9.2 compiler
Language files
Associate C and C++ files to Dev-C++
Shortcuts
Description
Position your mouse over a component to see its description.
Space required: 346.8MB
Nullsoft Install System v2.46
< Back
Next >
Cancel



ثم الضغط على زر Install.

يطلب برنامج التنصيب أن يتم تنزيل ملفات lex و bison في المسار: C:\GnuWin32 ويجب الالتزام بهذا الأمر لأننا سنحتاج لتوحيد العمل على كل الأجهزة والمسارات جزء من العمل.

برنامج DevC++ يجب تعديل مسار التنصيب وتنصيبه ضمن المسار: C:\Dev-Cpp سنحتاج منه المترجم GC++ الذي من خلاله سنقوم بترجمة ملفاتنا لبناء المترجم النهائي.

سنقوم بإنشاء مجلد داخل القرص C بالاسم التالي: Lex_Yacc وبداخله مجلد باسم examples وسنقوم بتنفيذ برامجنا داخل هذا المجلد اعتباراً من الجلسة القادمة.

انتهت الجلسة – د. علي ميا ، م. رشا شباني