






Material per la pràctica



Material al Campus Virtual

- Material disponible en Campus Virtual

Software

-  [CrossVisions Public 3.3 - Setup.exe \(interpret de CoSeL\)](#)
-  [CrossVisions Public 3.4 - Setup.exe \(interpret de CoSeL\)](#)
-  [CrossVisions.chm \(Tutorial CoSeL 3.4\)](#)
-  [NotePad++ CoSeL Coloreado.xml](#)
-  [Com.csm](#)

Projecte Pràctic

-  [Assignació ampliacions de LOOS per les pràctiques.pdf](#)
-  [Enunciado Practica Compiladors 2020-2021 V4.pdf](#)
-  [Base per fer el lliuraments de sintàctic: Practica Sintactic V1.zip](#)
-  [Lliurament sintàctic de la pràctica](#)

Fitxers: Practica Sintactic V?.zip

- Fitxers:
 - Com.csm: mòdul generador de compiladors per CoSeL
 - Int.csl: fitxer per on comença l'execució
 - PracticasLOOS.cip Fitxer de projecte per executar la pràctica (obre el CVIDE amb CoSeL)
 - Sintactic.csl: Fitxer de la pràctica

- Identificar-se

```
// =====  
// Identificació dels alumnes del grup =====  
// =====  
  
SetComponentsGrup([  
    // ("nom","cognoms","NIU","e-mail")  
    ("nom","cognoms","0000000","0000000@uab.cat")  
])
```

Sintactic.csl: Gramàtica

BNF_PARSER <programa>

TERMINALS

+ - * / identificador numero caracter string () =

...

// Declarar els nous símbols terminals que siguin necessaris

BNF

// <Programa> =====

Rule <programa>::=

{

<DecFun>

| <DecProc>

| <DecConstructor>

| <DecDestructor>

| <DecVar>

| <DecTipus>

| <DecClasse>

| IMPRIMIR_TAULA_DE_SIMBOLS ";"

}

...

// Modificar les regles o afegir noves regles

END;

Sintactic.csl: Informe

- Controlar el contingut dels informes de correcció

```
SetOpcionesInformes(  
    VerSintactico=>true,    // veure resultats de l'anàlisi sintàctica  
    VerSemantico=>false,    // veure resultats de l'anàlisi semàntica  
    VerGenCod=>false,       // veure resultats de la generació de codi  
    VerObligatorio=>true,   // veure resultats de les proves obligatòries  
    VerAdicional=>true,     // veure resultats de les proves addicionals  
    VerCorrecto=>true,      // veure les proves correctes  
    VerExtensions=>unbound // llista amb les extensions que volem veure  
);
```

- Per només veure el resultat per unes extensions concretes

```
VerExtensions=>["ArrayMultidimensional", "ArrayRangos", "DeclaracionVariableAuto", "VarInitSimple",  
    "OperadorBoolBits", "OperadorAbs", "OperadorMax", "OperadorMin", "OperadorPow", "OperadorDivMod"]
```

Init.csl

- Fitxer que volen executar de la pràctica
load Sintactic
//load Semantic
//load GenCod
- Prova que volem fer localment
Compila("LA MEVA PROVA DEL COMPILADOR",InStrStream("\{
Codi LOOS a compilar
\}"),true,false);
- Veure el arbre sintàctic: GraficArbreSintactic()
- La gramàtica compilada es guardar a la variable ParserGrammar
 - Després podem aplicar els mètodes de GramaticaBNF per analitzar-la.

CORRECCIÓ AMB LA WEB DE CORRECCIÓ

Quines ampliacions ha de fer cada alumne?

- Cada estudiant te 7 ampliacions a fer. NO HA DE FER LES 32 AMPLIACIONS DE L'ENUNCIAT.
- Projecto base per fer la pràctica

Arxiu per proves



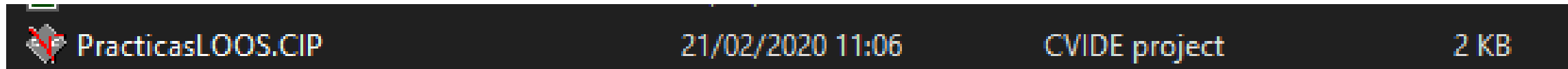
	Com.csm	20/02/2020 18:11	Archivo CSM	85 KB
	GenCod.csm	21/02/2020 11:04	Archivo CSM	48 KB
	init.csl	21/02/2020 10:56	Archivo CSL	1 KB
	PracticasLOOS.CIP	21/02/2020 11:06	CVIDE project	2 KB
	Semantic.csm	14/02/2020 10:50	Archivo CSM	46 KB
	Sintactic.csl	21/02/2020 10:59	Archivo CSL	12 KB



Únic arxiu que heu de entregar

Enunciado práctica

- Cóm editar el fixer Sintactic.csl?
 - Usant l'editor del CVIDE
 - Usant editores comunes de texto (Notepad++ i “NotePad++ CoSeL Coloreado.xml”, Bloc de Notas, el editor de programació qque vulgueu)
- Cóm comprobar que compila?
 - Doble clic a arxiu PracticasLOOS.CIP



- IMPORTANT!!!! Modificar primer Sintactic.csl para posar les dades de l'alumne

```
// =====  
// Identificació dels alumnes del grup =====  
// =====  
  
SetComponentsGrup([  
    // ("nom", "cognoms", "NIU", "e-mail")  
    ("nom", "cognoms", "00000000", "00000000@uab.cat")  
])
```

Enunciado práctica

- Cómo comprobar el progreso a la práctica?
 - Web de corrección de prácticas: <http://compiladors.uab.es>

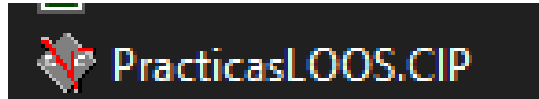


The image shows a web interface for a university compiler correction system. It has a blue header bar with the text "Corrector de la universidad". Below the header, there are two input fields: one for "Niu:" with the placeholder text "Introduce tu niu", and another for "Contraseña:" with the placeholder text "Contraseña". Below these fields is a blue button labeled "Enviar". At the bottom of the form, there is a green-bordered box containing the text: "Si todavía no has solicitado tu contraseña de acceso pulse en el siguiente enlace [aquí](#)".

Enunciado práctica

- CONSELL PEL US DEL CORRECTOR WEB

- Primer comprobar si compila en local



- Després pujar la pràctica (sintactic.csl)
- Si compila, et retorna un informe de correcció

Información entrega

Información asignatura
Nombre: Compiladors2020
Código: 2020

Información práctica
Nombre: Anàlisi Sintàctica
Fecha de entrega: 2020-03-08 23:59:00.0

Información última entrega
Informe: [Informe](#)
Entrega: [Script](#)

Archivo: Ningún archivo seleccionado

Ya has realizado una entrega anterior, si añades una nueva entrega esta quedará eliminada. ×

Informe de correcció

Resultados sintactico

```
-----  
Test sintactico obligatorio OK.....: 120 (70.5882%) <<<FALLA>>>  
Test sintactico adicional OK.....: 129 (68.984%)  
NOTA SINTACTICO.....: 0  
TEST OBLIGATORIOS QUE FALLAN (50):  
    SintacticoPruebaInstruccionForDowntoSimple1  
    SintacticoPruebaInstruccionForDowntoSimple2  
    SintacticoPruebaInstruccionForDowntoExp SintacticoInstruccionCase  
    SintacticoInstruccionCaseSinELSE  
SintacticoInstruccionCaseErrorSintactico1  
    SintacticoInstruccionCaseErrorSintacticoExpresion  
    SintacticoInstruccionCaseChar SintacticoInstruccionSelectConElse  
    SintacticoInstruccionSelectSinElse SintacticoOperadorNotBit  
    SintacticoOperadorOrBit SintacticoOperadorAndBit  
SintacticoOperadorAndOrBit  
    SintacticoOperadorAndOrNotBit SintacticoOperadorAbs1  
SintacticoOperadorAbs2
```

Test obligatoris:

- Es proporciona el codi del test que falla i perquè

Test adicionales:

- No se indica ni codi del test que falla ni perquè
- Nomes indica quants superats i quants fallen.

Exemples Ampliacions Sintàctiques per LOOS

OPERADOR FACTORIAL

Afegir l'operador factorial a LOOS

- Volem afegir l'operador factorial a LOOS amb la següent sintaxis

a=3!

b=(c+d)!! // Es pot fer el factorial del factorial

x=-3! // Primer fer factorial i després el canvi de signe

f(x,y)!

- Com es un operador afecte a les regles d'expressió. **On ho posem i Com?**

Rule <Expressio> ::= <TerBool> { "||" <TerBool> }

Rule <TerBool> ::= <FacBool> { "&&" <FacBool> }

Rule <FacBool> ::= <ExpArit> [(== | != | ">" | "<" | >= | <=) <ExpArit>]

Rule <ExpArit> ::= <terme> { (+ | -) <terme> }

Rule <terme> ::= <factor> { (* | /) <factor> }

Rule <factor> ::=

! <factor> | - <factor> | & <factor>

| "(" <Expressio> ")"

| Numero | Caràcter | String | true | false | Null

| "new" <tipus> [<Parametres>]

| this <PostFixe>

| Identificador [<Parametres>] <PostFixe>

Rule <Postfixe> ::= <Acces> [= <Expressio>]

Rule <Acces> ::= {
 "[" <Expressio> "]"
 | . identificador [<parametres>]
 | ^ }
Rule <parametres> ::=

"(" [<Expressio> { , <Expressio> }] ")"

Afegir l'operador factorial a LOOS: Proves

- Com es un operador afecte a les regles d'expressió. **On ho posem i Com?**
 - Com el factorial es posfixe el posarem a la regla <postfixe> on estan els operadors postfixes
 - On el posem

1.Rule <Postfixe> ::= {!} <Acces> [= <Expressio>]

2.Rule <Postfixe> ::= <Acces> {!} [= <Expressio>]

3.Rule <Postfixe> ::= <Acces> [= <Expressio> {!}]

4.Rule <Postfixe> ::= <Acces> [= <Expressio>] {!}

Afegir l'operador factorial a LOOS: Solució

- Volem afegir l'operador factorial a LOOS amb la següent sintaxis

a=3!

b=(c+d)!! // Es pot fer el factorial del factorial

x=-3! // Primer fer factorial i després el canvi de signe

f(x,y)!

- Com es un operador afecte a les regles d'expressió. **On ho posem i Com?**

Rule <factor> ::=

! <factor> | - <factor> | & <factor>
| "(" <Expressio> ")" {!}
| Numero {!} | Caràcter | String | true | false | Null
| "new" <tipus> [<Parametres>]
| this <PostFixe>
| Identificador [<Parametres>] <PostFixe>

Rule <Postfixe> ::= <Acces> [= <Expressio> | ! {!}]

Rule <Acces> ::= {
 "[" <Expressio> "]"
 | . identificador [<parametres>]
 | ^ }
Rule <parametres> ::=
 "(" [<Expressio> {, <Expressio> }] ")"

AFEGIR GOTO ESTIL ADA A LOOS

Afegir la instrucció goto i etiquetes a LOOS al estil de llenguatge ADA

- El llenguatge ADA es diu així en honor de Augusta Ada King, Contessa de Lovelace que va ser la primera programadora (1815-1852)
- ADA es un llenguatge de programació que dona un fort suport a l'enginyeria del software.
- Permet l'ús de goto però de forma que faci mal els ulls dels que revisen el codi
 - goto *etiqueta*;
 - <<*etiqueta*>>

- Exemple de goto afegit a LOOS

```
While i<10 do {  
    if a[i]<0 then goto negatiu;  
    i=i+1;  
}  
PrintLn "negatiu no trobat";  
goto NoNegatiu;  
<<Negatiu>>  
    PrintLn "S'ha trobat un negatiu a ", i;  
<<NoNegatiu>>
```

Goto i Etiquetes son instruccions

- Les instruccions a LOOS:

Rule <instruccio> ::=

<Expressio> ";"

| return [<Expressio>] ";"

| if <Expressio> then <instruccio> [else <instruccio>]

| while <Expressio> do <instruccio>

| Print <Expressio> {,<Expressio>} ";"

| Exit ";"

| delete <expressio>

| <Bloc>

| ";"

| IMPRIMIR_TAULA_DE_SIMBOLS ";"

Goto i Etiquetes: Solució

Rule <instruccio> ::=

| <Expressio> ";"

| return [<Expressio>] ";"

| if <Expressio> then <instruccio> [else <instruccio>]

| while <Expressio> do <instruccio>

| Print <Expressio> {,<Expressio>} ";"

| Exit ";"

| delete <expressio>

| <Bloc>

| ";"

| IMPRIMIR_TAUULA_DE_SIMBOLS ";"

| goto identificador ";"

| "<<" identificador ">>"

Declarar els nous símbols terminals

BNF_PARSER <programa>

TERMINALS

+ - * / identificador numero character string

() = == != > < >= <= ^

! && || , ; : . & "::" "[" "]" "{" "}"

Type Class Var Function Procedure constructor destructor

Integer Real Character Bool array of

if then else while do return Print Exit delete new

true false null this operator

IMPRIMIR_TAUULA_DE_SIMBOLS

goto "<<" ">>"

BNF

IF DEL LENGUATGE C A LOOS

Instrucció if del llenguatge C

- Volem afegir a LOOS la instrucció if al estil de C perquè no ens agrada el if then que ja té. Mantindrem l'if then de LOOS per compativilitat

- Exemple:

- A LOOS sense l'ampliació:

```
function maxim(a:integer,b:integer):integer
{
    if a>b then return a; else return b;
}
```

- A LOOS amb l'ampliació:

```
function maxim(a:integer,b:integer):integer
{
    if (a>b) return a; else return b;
}
```


Instruccions de LOOS

- Les instruccions a LOOS:

Rule <instruccio> ::=

<Expressio> ";"

| return [<Expressio>] ";"

| if <Expressio> then <instruccio> [else <instruccio>]

| while <Expressio> do <instruccio>

| Print <Expressio> {,<Expressio>} ";"

| Exit ";"

| delete <expressio>

| <Bloc>

| ";"

| IMPRIMIR_TAULA_DE_SIMBOLS ";"

Afegir l'if de C: Primer intent

Rule <instruccio> ::=

```
<Expressio> ";"  
| return [<Expressio>] ";"  
| if <Expressio> then <instruccio> [else <instruccio>]  
| if "(" <Expressio> ")" <instruccio> [else <instruccio>]  
| while <Expressio> do <instruccio>  
| Print <Expressio> {,<Expressio>} ";"  
| Exit ";"  
| delete <expressio>  
| <Bloc>  
| ";"  
| IMPRIMIR_TAULA_DE_SIMBOLS ";"
```

- On falla?

Afegir l'if de C: Segon intent

Rule <instruccio> ::=

```
<Expressio> ";"  
| return [<Expressio>] ";"  
| if ( "(" <Expressio> ")" | <Expressio> then )  
    <instruccio> [else <instruccio>]  
| while <Expressio> do <instruccio>  
| Print <Expressio> {,<Expressio>} ";"  
| Exit ";"  
| delete <expressio>  
| <Bloc>  
| ";"  
| IMPRIMIR_TAUULA_DE_SIMBOLS ";"
```

- On falla o pot fallar?
- Ho podem solucionar?

EXPRESSIÓ CONDICIONAL

Expressió condicional a LOOS

- Al llenguatge CoSeL es poden escriure expressions condicionals com:

```
Fun maxim(a,b,c)=> {  
    var tmp=if (a>b) a else b;  
    return if (tmp>c) tmp else c;  
}
```

- Volem fer el mateix a LOOS

```
Function maxim(a:integer,b:integer,c:integer):integer  
{  
    var tmp:integer;  
    tmp=if a>b then a else b;  
    return if tmp>c then tmp else c;  
}
```

Expressió condicional a LOOS

- Altres exemples

```
tmp=10+if a>b then a else b;
```

```
tmp=10+(if a>b then a else b);
```

```
tmp=10+(if a>b then a*5 else (b*6+5));
```

```
tmp=10+if a>b then if c>a then c else a else if b>c then b else c;
```

```
tmp=10+(if a>b then (if c>a then c else a) else (if b>c then b else c));
```

- On posar l'expressió condicional

- Com es un operador va a les regles d'expressió.
- Te més prioritat que tots els altres operadors que tingui a la seva esquerra i menys prioritat que tots els operador que tingui a la seva dreta.

Expressió condicional

- A quina regla va l'expressió condicional?

Rule <Expressio> ::= <TerBool> { "||" <TerBool> }

Rule <TerBool> ::= <FacBool> { "&&" <FacBool> }

Rule <FacBool> ::= <ExpArit> [(== | != | ">" | "<" | >= | <=) <ExpArit>]

Rule <ExpArit> ::= <terme> { (+ | -) <terme> }

Rule <terme> ::= <factor> { (* | /) <factor> }

Rule <factor> ::=

! <factor> | - <factor> | & <factor>
| "(" <Expressio> ")"
| Numero | Caràcter | String | true | false | Null
| "new" <tipus> [<Parametres>]
| this <PostFixe>
| Identificador [<Parametres>] <PostFixe>

Rule <Postfixe> ::= <Acces> [= <Expressio>]

Rule <Acces> ::= {
| "[" <Expressio> "]"
| . identificador [<parametres>]
| ^ }

Rule <parametres> ::=
"(" [<Expressio> { , <Expressio> }] ")"

Expressió condicional

- A quina regla va l'expressió condicional?

Rule <Expressio> ::= <TerBool> { "||" <TerBool> }

Rule <TerBool> ::= <FacBool> { "&&" <FacBool> }

Rule <FacBool> ::= <ExpArit> [(== | != | ">" | "<" | >= | <=) <ExpArit>]

Rule <ExpArit> ::= <terme> { (+ | -) <terme> }

Rule <terme> ::= <factor> { (* | /) <factor> }

Rule <factor> ::=

! <factor> | - <factor> | & <factor>
| "(" <Expressio> ")"
| Numero | Caràcter | String | true | false | Null
| "new" <tipus> [<Parametres>]
| this <PostFixe>
| Identificador [<Parametres>] <PostFixe>
| if <expressio> then <expressio> else <expressio>

- Funciona correctament?

```
Rule <Postfixe> ::= <Acces> [ = <Expressio> ]
Rule <Acces> ::= {
    "[" <Expressio> "]"
    | . identificador [<parametres>]
    | ^ }
Rule <parametres> ::=
    "(" [<Expressio> { , <Expressio> } ] ")"
```


Problema amb l'expressió condicional

- A Instrucció una expressió pot començar amb if i if then igual. Això fa que el analitzador no accepti la instrucció if.

Rule <instruccio> ::=

```
<Expressio> ";" // Pot començar amb if a <factor> ::= ... | if ...  
| return [<Expressio>] ";"  
| if <Expressio> then <instruccio> [else <instruccio>] // Comença amb if  
| while <Expressio> do <instruccio>  
| Print <Expressio> {,<Expressio>} ";"  
| Exit ";"  
| delete <expressio>  
| <Bloc>  
| ";"  
| goto identificador ";"  
| "<<" identificador ">>"
```

- Solució afegir una nova expressió que contingui if i substituir expressió per aquesta on no generi ambigüitat

Lliurament Parcial d'Anàlisi Sintàctica per LOOS

Problema a Lliurar

Compte com problema i no com a pràctica

Lliurament Parcial d'Anàlisi Sintàctica

- De les extensions que os han tocat fer per la pràctica heu de fer les següents en aquest lliurament parcial (de les següents ampliacions heu de fer **les tres que teniu assignades**):
 - ArrayMultidimensional
 - ArrayRangos
 - DeclaracionVariableAuto
 - VarInitSimple
 - OperadorBoolBits
 - OperadorAbs
 - OperadorMax
 - OperadorMin
 - OperadorPow
 - OperadorDivMod
- Al campus virtual, a l'apartat de projecte pràctic trobareu les ampliacions/extensions que teniu assignades
 - Assignació ampliacions de LOOS per les pràctiques.pdf

Base per fer aquest problema

- Per utilitzar CoSeL heu d'instal·lar
 - CrossVisions Public 3.4 - Setup.exe
- Els fitxers que necessiteu els trobareu al campus virtual al apartat del projecte pràctic
 - Base per fer el lliuraments de sintàctic: Practica Sintactic V1.zip
- El lliurament es fa al campus virtual a l'apartat Tema 2. Anàlisi Sintàctica.
- Podeu utilitzar el corrector automàtic per comprovar que tot funciona correctament
 - Web: <http://compiladors.uab.es>
 - Ja esteu donat d'alta.
 - El password el podeu obtenir demant-lo a la mateixa web. L'envia per e-mail al vostre e-mail institucional.
 - Per nomes veure a l'informe els test de les ampliacions a lliurar pode afegir al fitxer sintàctic.csl substituint VerExtensions=> unbound per
 - VerExtensions=>["ArrayMultidimensional", "ArrayRangos", "DeclaracionVariableAuto", "VarInitSimple", "OperadorBoolBits", "OperadorAbs", "OperadorMax", "OperadorMin", "OperadorPow", "OperadorDivMod"]