UNIVERSIDADE
DE TRÁS-OS-MONTES
E ALTO DOURO

Título do Relatório - Etapa

Engenharia Informática

Compiladores

Arsénio Reis

Teresa A. Perdicoulis

Autores:

Nuno Ferreira al76889

Francisco Ramos al76926

Gonçalo Costa al76131

Vila Real, 2022

1. Introdução:

Um analisador léxico é um programa que permite ler um conjunto de caracteres (exemplo: ficheiros de texto ou programa-fonte) e produzir uma sequência de componentes léxicos.

O LEX/FLEX é uma ferramenta que permite gerar analisadores léxicos. Estes analisadores são capazes de reconhecer padrões léxicos em texto.

Um programa em FLEX é constituído por três secções: declarações, regras e rotinas auxiliares. A separação entre as secções é feita inserindo uma linha com o símbolo "%%".

2. Tarefas:

Tarefa A- Analisador léxico utilizando LEX/FLEX

1. **Identificar o seguinte conjunto de instruções que podem ser usadas para manipular o robot Compilator:**

- Declaração de variáveis

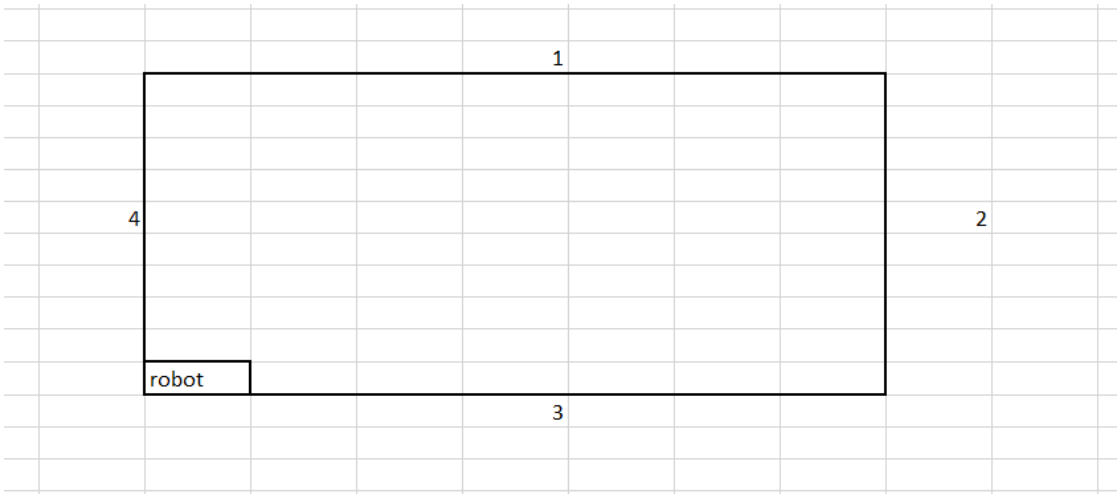
```
%option noyywrap
%x c_p
%{
        int posicaox = 1, posicaoy = 1, direcaoR = 1, PosicaoPinca = 1, EstadoP = 2, Varaux = 0;
        void Mostrardirecao();
        void Serro();
        void Andar();
        void MostrarPosicao();
        void ManipularString();
        void Rodar();
        void ControloPinca();
        void MostrarPosicaoPinca();
        void MostrarEstadoPinca();
%}
%%
%%
```

- VIRAR-DIREITA: Indica que o robot que deve rodar 90º à direita. Sempre que o robot atinge a posição 5 volta à posição 1.

```
VIRAR-DIREITA {

                direcaoR++;
                if(direcaoR == 5){ direcaoR = 1;}
}
```

Exemplo:



- VIRAR-ESQUERDA: Indica o robot que deve rodar 90º à esquerda. Ao contrário do anterior sempre que atingir a posição 0 volta à posição 4.

```
VIRAR-ESQUERDA {

                direcaoR--;
                if(direcaoR == 0){ direcaoR = 4;}
}
```

- ANDAR(N): Indica ao robot que deve deslocar em frente N posições, onde N é um número inteiro entre 1 e 100.

```
ANDAR\((100|[1-9][0-9]|[1-9])\) {
                    ManipularString();
                    Andar();
}
```

- Função usada para a instrução ANDAR(N). Neste caso criamos uma variável auxiliar para guardar os números que estavam dentro de parênteses. E por fim para passamos a variável aux de string para inteiro.

```
void ManipularString()
{
char aux[4]="";
int i = 0 ;
        Varaux = 0;

        if(yytext[6] == '-'){
        i = 1;
        aux[0] = yytext[6];
        }

        if(yytext[7 + i] == ')'){
        aux[0 + i] = yytext[6 + i];
        }

        if(yytext[8 + i] == ')'){
        aux[0 + i] = yytext[6 + i];
        aux[1 + i] = yytext[7 + i];
        }

        if(yytext[9 + i] == ')'){
        aux[0 + i] = yytext[6 + i];
        aux[1 + i] = yytext[7 + i];
        aux[2 + i] = yytext[8 + i];
        }
        Varaux = atoi(aux);
}
```

Exemplo:

0	1	2	3	4	5	6	7	8	9	10
A	N	D	A	R	(	0	)			
A	N	D	A	R	(	1	5	)		
A	N	D	A	R	(	1	0	0	)	

- PINÇA(P): Indica ao robot se deve abrir ou fechar a pinça, onde P assume valores de “ABRIR” ou “Fechar”. Neste caso, quando o valor da pinça for igual a 1, a pinça já se encontra aberta. Quando assume o valor de 2, a pinça já se encontra fechada.

```
"PINCA("    BEGIN(c_p);
<c_p>"ABRIR" {  if(EstadoP == 1)  printf("Erro, a pinca ja se encontra aberta"); else EstadoP = 1; }
<c_p>"FECHAR"  { if(EstadoP == 2) printf("Erro, a pinca ja se encontra fechada"); else EstadoP = 2;  }
<c_p>")"    BEGIN(INITIAL);
```

- RODAR(G): indica ao robot que deve rodar o braço em G graus, onde G pode assumir valores positivos ou negativos, múltiplos de 45, com um máximo de 360.

```
RODAR\((-)?(45|90|135|180|270|315|360)\) {

                ManipularString();
                Rodar();
                ControloPinca();
}
```

- Função usada para a instrução RODAR(G). Esta função é a mesma utilizada na instrução ANDAR(N), só que na função RODAR(G) pode apresentar valores negativos.

```
void ManipularString()
{
char aux[4]="";
int i = 0 ;
        Varaux = 0;

        if(yytext[6] == '-'){
        i = 1;
        aux[0] = yytext[6];
        }

        if(yytext[7 + i] == ')'){
        aux[0 + i] = yytext[6 + i];
        }

        if(yytext[8 + i] == ')'){
        aux[0 + i] = yytext[6 + i];
        aux[1 + i] = yytext[7 + i];
        }

        if(yytext[9 + i] == ')'){
        aux[0 + i] = yytext[6 + i];
        aux[1 + i] = yytext[7 + i];
        aux[2 + i] = yytext[8 + i];
        }
        Varaux = atoi(aux);
}
```

Exemplo:

0	1	2	3	4	5	6	7	8	9	10
R	O	D	A	R	(	0	)			
R	O	D	A	R	(	4	5	)		
R	O	D	A	R	(	3	6	0	)	
R	O	D	A	R	(	-	0	)		
R	O	D	A	R	(	-	4	5	)	
R	O	D	A	R	(	-	3	6	0	)

- Função de Controlo da Pinça.

```
void ControloPinca()
{
        if(PosicaoPinca == 9) {PosicaoPinca = 1;}
        if(PosicaoPinca == 10) {PosicaoPinca = 2;}
        if(PosicaoPinca == 11) {PosicaoPinca = 3;}
        if(PosicaoPinca == 12) {PosicaoPinca = 4;}
        if(PosicaoPinca == 0) {PosicaoPinca = 8;}
        if(PosicaoPinca == -1) {PosicaoPinca = 7;}
        if(PosicaoPinca == -2) {PosicaoPinca = 6;}
}
```

Exemplo:

		1		
	8		2	
7				3
	6		4	
		5		

2. **Despoletar as ações correspondentes a cada uma das instruções, guardando, após cada instrução correta recebida, o estado atual do robot:**

- MostrarPosição(): Esta função serve para mostrar a posição atual do robot.

```
void MostrarPosicao()
{
        printf("A posicao do robot é %d x %d \n", posicaox, posicaoy);
}
```

- Mostrardirecao(): Esta função serve para mostrar a direção para onde o robot está virado.

```
void Mostrardirecao()
{
        if(direcaoR == 1){
                printf("O robot está virado para norte \n");
        }
        if(direcaoR == 2){
                printf("O robot está virado para este \n");
        }
        if(direcaoR == 3){
                printf("O robot está virado para sul \n");
        }
        if(direcaoR == 4){
                printf("O robot está virado para oeste \n");
        }
}
```

- MostrarEstadoPinca(): Esta função serve para mostrar o estado da pinça.

```
void MostrarEstadoPinca()
{
        if(EstadoP == 2) {
                printf("A pinca esta fechada \n");
        }else{
                printf("A pinca esta aberta \n");
        }
}
```

- MostrarPosicaoPinca(): Esta função permite-nos verificar para que ponto cardeal a pinça está virada.

```
void MostrarPosicaoPinca()
{
        if(PosicaoPinca ==1) {printf("A pinca esta virada para Norte \n ");}
        if(PosicaoPinca ==2) {printf("A pinca esta virada para Nordeste \n ");}
        if(PosicaoPinca ==3) {printf("A pinca esta virada para Este \n ");}
        if(PosicaoPinca ==4) {printf("A pinca esta virada para Sudeste \n ");}
        if(PosicaoPinca ==5) {printf("A pinca esta virada para Sul \n ");}
        if(PosicaoPinca ==6) {printf("A pinca esta virada para Sudoeste \n ");}
        if(PosicaoPinca ==7) {printf("A pinca esta virada para Oeste \n ");}
        if(PosicaoPinca ==8) {printf("A pinca esta virada para Nordeste \n ");}
}
```

3. Validar situações irregulares, e lançar um alerta quando estas ocorrerem, nomeadamente

- Após uma instrução ANDAR(N), o robot ficaria de fora dos limites do plano.

```
void Andar()
{
        if(direcaoR == 1){
                posicaoy = posicaoy + Varaux;
                if ( posicaoy > 100){
                Serro();
                posicaoy = posicaoy - Varaux;
                }
        }
        if(direcaoR == 2){
                posicaox = posicaox + Varaux;
                if ( posicaox > 100){
                Serro();
                posicaox = posicaox - Varaux;
                }
        }if(direcaoR == 3){
                posicaoy = posicaoy - Varaux;
                if ( posicaoy < 1){
                Serro();
                posicaoy = posicaoy + Varaux;
                }
        }if(direcaoR == 4){
                posicaox = posicaox - Varaux;
                if ( posicaox < 1){
                Serro();
                posicaox = posicaox + Varaux;
                }
        }
}
```

- Receber uma instrução RODAR(G) em que o número de graus faça com que o braço fique na mesma posição que estava antes de receber essa instrução.

```
void Rodar()
{
        if(Varaux == 0 || Varaux == 360) {
                PosicaoPinca = PosicaoPinca;
        }
        else if(Varaux == 45 || Varaux == -315) {
                PosicaoPinca = PosicaoPinca + 1;
        }
        else if(Varaux == 90 || Varaux == -270) {
                PosicaoPinca = PosicaoPinca + 2;
        }
        else if(Varaux == 135 || Varaux == -225) {
                PosicaoPinca = PosicaoPinca + 3;
        }
        else if(Varaux == 180 || Varaux == -180) {
                PosicaoPinca = PosicaoPinca + 4;
        }
        else if(Varaux == 225 || Varaux == -135) {
                PosicaoPinca = PosicaoPinca - 3;
        }
        else if(Varaux == 270 || Varaux == -90) {
                PosicaoPinca = PosicaoPinca - 2;
        }
        else if(Varaux == 315 || Varaux == -45) {
                PosicaoPinca = PosicaoPinca - 1;
        }
        else{
                printf("O comando introduzido nao altera a posicao da pinca. \n");
                Serro();
        }
}
```

- Receber uma instrução PINÇA(ABRIR) quando a pinça já se encontra aberta ou uma instrução PINÇA(FECHAR) quando esta já se encontra fechada.

```
"PINCA("    BEGIN(c_p);
<c_p>"ABRIR" {  if(EstadoP == 1)  printf("Erro, a pinca ja se encontra aberta"); else EstadoP = 1; }
<c_p>"FECHAR"  { if(EstadoP == 2) printf("Erro, a pinca ja se encontra fechada"); else EstadoP = 2;  }
<c_p>")"    BEGIN(INITIAL);
```

- Função para erros:

```
void Serro()
{
        printf("Erro no comando %s, nao foi efetuado nenhuma operacao \n",yytext);
}
```

4. **Imprimir o estado final do robot (posição atual, direção para onde está virado, posição do braço e estado da pinça)**

```
int main()
{
        yylex();
        Mostrardirecao();
        MostrarPosicao();
        MostrarPosicaoPinca();
        MostrarEstadoPinca();

}
```

Tarefa B- Crie um ficheiro de texto para teste, que contenha a sequência de instruções necessárias para que o robot termine no seguinte estado:

- Posição (x , y) = (40 , 60)
- Direção do robot = +*yy* (Norte)
- Posição do braço = -*yy* (Sul)
- Estado da pinça: Aberta

Conteúdo no Ficheiro teste.txt e o resultado do conteúdo do ficheiro teste.txt

```
VIRAR-DIREITA
ANDAR(39)
VIRAR-ESQUERDA
ANDAR(59)
RODAR(180)
PINCA(ABRIR)
~
```

```
goncalo@LAPTOP-GON5EMI9:~$ vi trabalho.l
goncalo@LAPTOP-GON5EMI9:~$ vi teste.txt
goncalo@LAPTOP-GON5EMI9:~$ lex trabalho.l
goncalo@LAPTOP-GON5EMI9:~$ cc lex.yy.c
goncalo@LAPTOP-GON5EMI9:~$ ./a.out <teste.txt
O robot está virado para norte
A posicao do robot é 40 x 60
A pinca esta virada para Sul
 A pinca esta aberta
goncalo@LAPTOP-GON5EMI9:~$
```