C Translator

Rafael Caro de los Reyes

Alejandro García Ramos

Adrián Muñoz López

INTRODUCTION

src

tests

documentación

README.md

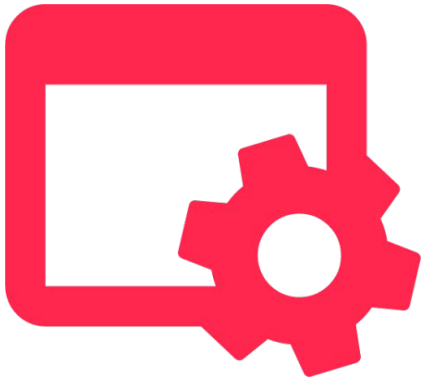
HOW TO USE IT?

$ python3 src/main.py traducir.c traducido.s

```
documentacion  makefile  parser.out  README.md  src  tests  traduccion.s
```

Extra information:

Easy to use:
Terminal interface with makefile based automated tests.



ANSI C standard

Project modules

Project milestones

CParser and asm generation

Accomplished
- Local and global variable declarations. - Arithmetic and logical operations. - Function calls and declarations. - Conditional and loop blocks. - Pointers, pointer arithmetics and multidimensional arrays. - Printf and scanf variadic functions.

Not accomplished
- Variable initialization on declaration not allowed. - Array initialization on declaration not allowed. - Variables are not permitted in array index, only constant numbers are allowed - In function calls you can only use either variables or numeric constants. - You cannot write multiple ‘;’ in a row.

Tests:

Algebra , variables and globals

Everything works fine except initializing variables at declaration

All declarations must be declared at the start of the function

```
int a,b;
int c;

int main() {
    int x,y,z;
    //int var = 20; <-- Not possible

    x = 10;
    z = y = x;

    x = y*(z+x)-10;
    c = x;

    return 0;
}
```

Printf,Scanf and Functions

you cannot call a function using variables and numbers at the same time

```c
int f(int a,int b ,int c){
  return a+c+b;
}

int f2(int a){
  return a;
}

void f3(){}

int main() {
  int a,c;

  a = f(5,1,4);
  c = f2(a);
  //c = f2() invalid wrong number of parameters
  //c = f3() invalid f3 is void
  scanf("insert number %i",&a);
  printf("this is a print : %i",a);

    return 0;
}
```

arrays, matrices and pointers

Not possible to use variables as indexes

```
int main() {
   int mat4[5][5][5][5];
   int mat3[5][5][5];
   int mat2[5][5];
   int mat1[5];
   int* punt;
    int a;
   punt = &a;

   mat1[1] = 10;
   mat3[2][3][4] = 20;
   mat2[2][1] = mat1[1];

    //mat1[a] = a; cant use varibles as indexes
   //mat4[2][2] = 10; wrong dimension
   //mat2[1000][3]; wrong, out of range
   return 0;
}
```

IF , ELSE AND WHILE

Works fine,
we didn't find any errors

```c
int main() {
    int x;

    x = 20;

    if (x >= 10 && x <= 30) {
        if (x <= 30) {
            printf("x está entre 10 y 30 \n");
        } else {
            printf("x es mayor o igual a 30 \n");
        }
    } else {
        printf("x es menor o igual a 10 \n");

        while(x <= 100){
          printf("x = %i\n",x);
          x = x +1;
        }
    }

    return 0;
}
```

All of the previous C programs can be translated into their corresponding assembly code.

```c
int main() {
    int x;

    x = 20;

    while(x<=30){
      printf("inside if");
      x = x + 1;
    }

    return 0;
}
```

```
.Section_rodata
.S1:    .text "inside if"
.end_rodata

.text
.globl main
.type main, @function
main:
pushl %ebp
movl %esp, %ebp
subl $8, %esp

movl $20, %eax
movl %eax, -4(%ebp)

while2_ini:
movl -4(%ebp), %eax
movl $30, %ebx
cmpl %ebx, %eax
movl $1, %eax
jg verdadero3

movl $0, %eax
verdadero3:
cmpl $0, %eax
jne while2_fin

pushl s1
call printf
addl $4, %esp

movl -4(%ebp), %eax
movl $1, %ebx
addl %ebx, %eax
movl %eax, -4(%ebp)
jmp while2_ini

while2_fin:
movl $0, %eax
movl %ebp, %esp
popl %ebp
ret
```

It can generate several hundreds lines of assembly!

```
786     movl $0, %eax
787     movl %ebp, %esp
788     popl %ebp
789     ret
790
```

```
int global1 [100][100];


int suma(int a, int b) {
    return a + b;
}

// Función que no devuelve nada
void mensaje () {
    printf ("Esta función no devuelve ningún valor    \n");
}

int f(int a, int b ,int c){
  int d;
  d = 10;
  return a+c+b+d;
}

int f2(int a){
  return a;
}

void f3(){}

int operacion_ridicula  (int a, int b, int c , int d,int f){
  int ret ;
  ret = a*b*(f/(d - a + 5 - c))-(93289329 );

  return ret;
}

void bucle_while_aburdo  (int i){
  while(i <= 100){
    while(i <= 100){
      while(i <= 100){
        while(i <= 100){
          while(i <= 100){
            while(i <= 100){
              while(i <= 100){
                while(i <= 100){
                  while(i <= 100){
                    i = i + 1;
                  }
                }
              }
            }
          }
        }
      }
    }
  }
}

int main() {
  int resultado ;
  int a,c;

  resultado  = suma(5, 10);
  printf ("Resultado de la suma:   %d\n", resultado );

  mensaje ();

  bucle_while_aburdo  (resultado );

  a = f(5,1,4);
  c = f2(a);
  c = operacion_ridicula  (1,23,4,56,67);
  scanf ("insert number   %i",&a);
  printf ("this is a print :   %i",a);

  return 0;
}
```