Relatório etapa 6 Compiladores:

Funções implementadas:
Declaração e atribuição de variáveis, comando if, while, return, read e print.
Expressões aritméticas de soma, subtração, divisão e multiplicação, e operações booleanas como NOT, AND, OR, EQUAL, menor, maior.

Funções não implementadas:
Declaração e chamada de funções com parâmetros e inicialização de vetores.
Também não foi implementado as funções de verificação semântica (etapa 4).

Teste 1:
Entrada:

```
int 5 = #1;
char x = 'F';

int main() {
  while (5 < #10) {
    print "5 =" 5;
    5 = 5 + #1;
  }
/**
print char printa o numero, mas
print literal printa a letra
**/
  print x 'R';
}
```

Saída:

```
5 =1
5 =2
5 =3
5 =4
5 =5
5 =6
5 =7
5 =8
5 =9
70
R
```

Código gerado:

```
.data
.printInt: .string "%d\n"
.printChar: .string "%c\n"
.globl main
stringName1:
.string "5 ="
stringName0:
.long 'F'
stringName2:
.long 'R'
c_1:
.long 1
c_10:
.long 10
_5:
.long 0
_x:
.long 0
temp0:
.long 0
```

```
temp1:
.long 0
temp2:
.long 0
temp3:
.long 0
.text
main:
movl c_1(%rip), %eax
 movl %eax, _5(%rip)
movl stringName0(%rip), %eax
 movl %eax, _x(%rip)
_main:
 pushq %rbp
 movq %rsp, %rbp
label3:
movl _5(%rip), %edx
 movl c_10(%rip), %eax
 cmpl %eax, %edx
 setl %al
 movzbl %al, %eax
 movl %eax, temp2(%rip)
movl temp2(%rip), %eax
 testl %eax, %eax
 je label2
leaq stringName1(%rip), %rax
 movq %rax, %rdi
 call printf@PLT
movl _5(%rip), %eax
 movl %eax, %esi
 leaq .printInt(%rip), %rax
 movq %rax, %rdi
 call printf@PLT
movl _5(%rip), %eax
 addl c_1(%rip), %eax
 movl %eax, temp3(%rip)
movl temp3(%rip), %eax
 movl %eax, _5(%rip)
jmp label3
label2:
movl _x(%rip), %eax
 movl %eax, %esi
 leaq .printInt(%rip), %rax
 movq %rax, %rdi
 call printf@PLT
movl stringName2(%rip), %eax
 movl %eax, %esi
 leaq .printChar(%rip), %rax
```

```
 movq %rax, %rdi
 call printf@PLT
popq %rbp
 ret
.section .note.GNU-stack,"",@progbits
```

Teste 2:
Entrada:

```
int 5 = #1;

int main() {
  if ((~5 | #0) = #0) then {
    print "verdadeiro";
  } else {
    print "falso";
  }
  read 5;
  return 5;
}
```

Saída: verdadeiro

Código gerado:

```
.data
.printInt: .string "%d\n"
.printChar: .string "%c\n"
.globl main
stringName1:
.string "falso"
stringName0:
.string "verdadeiro"
c_0:
.long 0
c_1:
.long 1
_5:
.long 0
temp0:
.long 0
temp1:
.long 0
temp2:
.long 0
temp3:
.long 0
temp4:
.long 0
temp5:
.long 0
```

```
.text
main:
movl c_1(%rip), %eax
 movl %eax, _5(%rip)
_main:
 pushq %rbp
 movq %rsp, %rbp
movl _5(%rip), %eax
 testl %eax, %eax
 sete %al
 movzbl %al, %eax
 movl %eax, temp3(%rip)
movl temp3(%rip), %edx
 movl c_0(%rip), %eax
 orl %edx, %eax
 movl %eax, temp4(%rip)
movl temp4(%rip), %edx
 movl c_0(%rip), %eax
 cmpl %eax, %edx
 sete %al
 movzbl %al, %eax
 movl %eax, temp5(%rip)
movl temp5(%rip), %eax
 testl %eax, %eax
 je label2
leaq stringName0(%rip), %rax
 movq %rax, %rdi
 call printf@PLT
jmp label3
label2:
leaq stringName1(%rip), %rax
 movq %rax, %rdi
 call printf@PLT
label3:
call getchar@PLT
 movl %eax, _5(%rip)
movl _5(%rip), %eax
popq %rbp
 ret
popq %rbp
 ret
.section .note.GNU-stack,"",@progbits
```

Teste 3:

Entrada:

```
int a = #2;
int b = #4;

int main() {
  a = a * (#5 + #2) / #2 - b;
  print a;
  b = #2;
}


int fun() {
  1 = #10;
  return 1;
}
```

Saída: 3

Código gerado:

```
.data
.printInt: .string "%d\n"
.printChar: .string "%c\n"
.globl main
stringName1:
.string "falso"
stringName0:
.string "verdadeiro"
c_0:
.long 0
c_1:
.long 1
_5:
.long 0
temp0:
.long 0
temp1:
.long 0
temp2:
.long 0
temp3:
.long 0
temp4:
.long 0
temp5:
.long 0
.text
main:
```

```
movl c_1(%rip), %eax
 movl %eax, _5(%rip)
_main:
 pushq %rbp
 movq %rsp, %rbp
movl _5(%rip), %eax
 testl %eax, %eax
 sete %al
 movzbl %al, %eax
 movl %eax, temp3(%rip)
movl temp3(%rip), %edx
 movl c_0(%rip), %eax
 orl %edx, %eax
 movl %eax, temp4(%rip)
movl temp4(%rip), %edx
 movl c_0(%rip), %eax
 cmpl %eax, %edx
 sete %al
 movzbl %al, %eax
 movl %eax, temp5(%rip)
movl temp5(%rip), %eax
 testl %eax, %eax
 je label2
leaq stringName0(%rip), %rax
 movq %rax, %rdi
 call printf@PLT
jmp label3
label2:
leaq stringName1(%rip), %rax
 movq %rax, %rdi
 call printf@PLT
label3:
call getchar@PLT
 movl %eax, _5(%rip)
movl _5(%rip), %eax
popq %rbp
 ret
popq %rbp
 ret
.section .note.GNU-stack,"",@progbits
```