Universidade Estadual de Santa Cruz – UESC

Relatório de Implementação do P-Code

Relatório de implementações realizadas por Yan Costa Macedo

Disciplina Compiladores.

Curso Ciência da Computação

Semestre 2022.2

Professor César Alberto Bravo Pariente

Ilhéus – BA
2022

SUMÁRIO

P-CODE – GITHUB

O Código fonte pode ser acessado a partir do link:
https://github.com/KiritoKi/P-Code-Machine

No repositório encontra-se o código-fonte assim como as entradas de cada exercício em .txt

RESOLUÇÕES DOS EXERCÍCIOS

Exercício 01 :

Soma de dois números inteiros;

Resolução:

```
...Starting the P-code...
Inst        Level           Arg             topstack        program         Stack
INT         0               4               0               1               0 0 0 0
LIT         0               6               6               2               0 0 0 0 6
LIT         0               4               4               3               0 0 0 0 6 4
OPR         0               2               10              4               0 0 0 0 10
STO         0               3               10              5               0 0 0 10
OPR         0               0               0               0               0
kirito@kirito-Aspire-A515-41G:~/Documentos/Compiladores/PCODE - 1$ _
```

Exercício 02:

Soma dos números naturais de 1 até 10;

Resolução:

```
➜  PCODE - 1 cd "/home/kirito/Documentos/Compiladores/PCODE - 1"
➜  PCODE - 1 ./"p-code"
 Please enter the instructions (Instruction END 0 0 to stopstack the input)
 INT 0 4
 LIT 0 10
 STO 0 3
 LOD 0 3
 LOD 0 3
 LIT 0 1
 OPR 0 3
 STO 0 3
 LOD 0 3
 JPC 0 13
 LOD 0 3
 OPR 0 2
 JMP 0 4
 STO 0 3
 OPR 0 0
 END 0 0
 ---End of Program---
```

```
OPR         0               2               55              12              0 0 0 1 55
JMP         0               4               55              4               0 0 0 1 55
LOD         0               3               1               5               0 0 0 1 55 1
LIT         0               1               1               6               0 0 0 1 55 1 1
OPR         0               3               0               7               0 0 0 1 55 0
STO         0               3               55              8               0 0 0 0 55
LOD         0               3               0               9               0 0 0 0 55 0
JPC         0               13              55              13              0 0 0 0 55
STO         0               3               55              14              0 0 0 55
OPR         0               0               0               0               0
➜  PCODE - 1 _
```

Exercício 03:

Soma dos números naturais de 1 até 100;

Resolução:

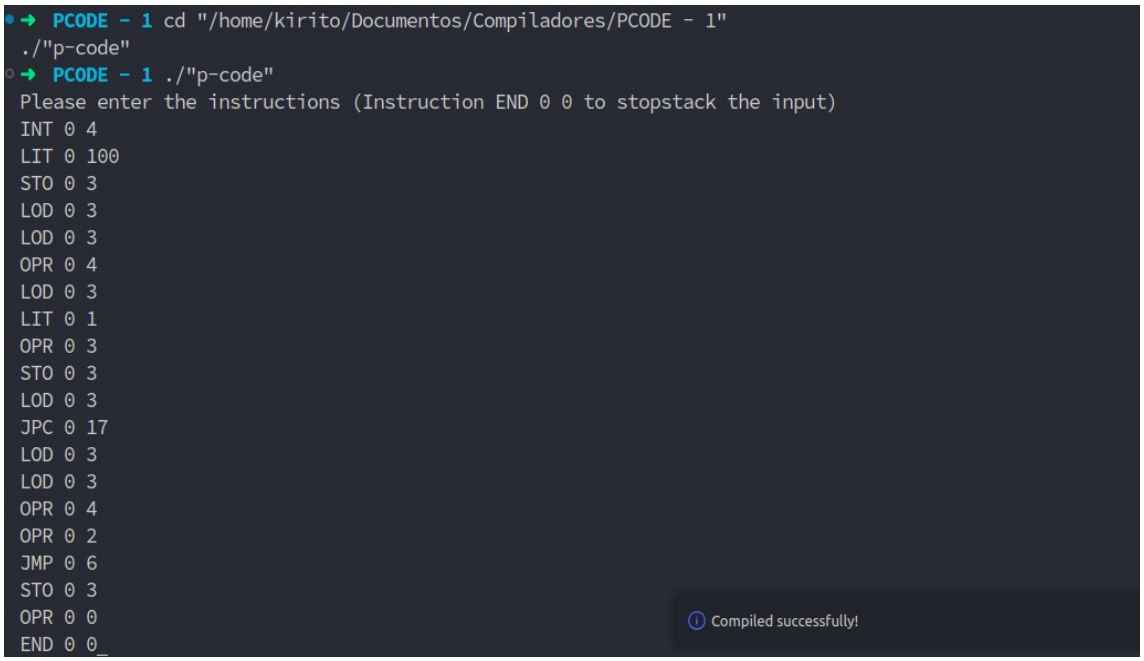
```
➜  PCODE - 1 cd "/home/kirito/Documentos/Compiladores/PCODE - 1"
 ./"p-code"
➜  PCODE - 1 ./"p-code"
 Please enter the instructions (Instruction END 0 0 to stopstack the input)
 INT 0 4
 LIT 0 100
 STO 0 3
 LOD 0 3
 LOD 0 3
 LIT 0 1
 OPR 0 3
 STO 0 3
 LOD 0 3
 JPC 0 13
 LOD 0 3
 OPR 0 2
 JMP 0 4
 STO 0 3
 OPR 0 0
 END 0 0
 ---End of Program---
```

```
 STO       0         3         5049      8         0 0 0 1 5049
 LOD       0         3         1         9         0 0 0 1 5049 1
 JPC       0         13        5049      10        0 0 0 1 5049
 LOD       0         3         1         11        0 0 0 1 5049 1
 OPR       0         2         5050      12        0 0 0 1 5050
 JMP       0         4         5050      4         0 0 0 1 5050
 LOD       0         3         1         5         0 0 0 1 5050 1
 LIT       0         1         1         6         0 0 0 1 5050 1 1
 OPR       0         3         0         7         0 0 0 1 5050 0
 STO       0         3         5050      8         0 0 0 0 5050
 LOD       0         3         0         9         0 0 0 0 5050 0
 JPC       0         13        5050      13        0 0 0 0 5050
 STO       0         3         5050      14        0 0 0 5050
 OPR       0         0         0         0         0
➜  PCODE - 1
```

Exercício 04:

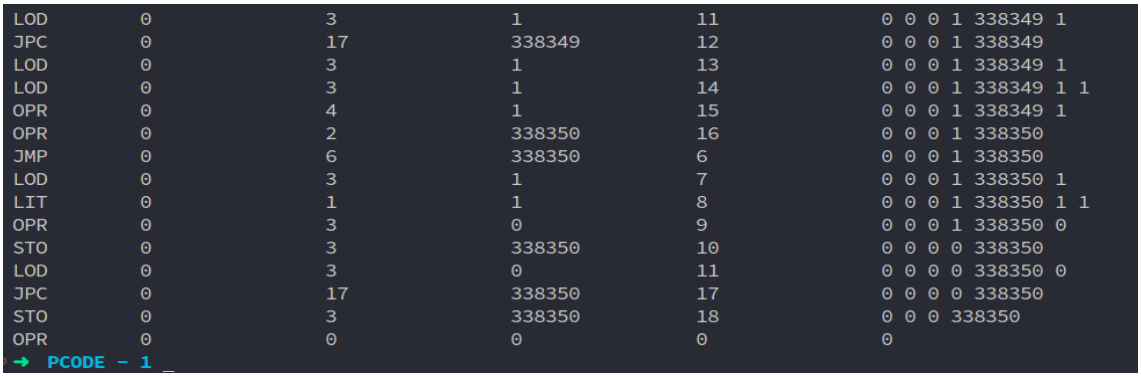
Soma dos quadrados dos números naturais de 1 até 100(iterativamente);

Resolução:



```
➜  PCODE - 1 cd "/home/kirito/Documentos/Compiladores/PCODE - 1"
 ./"p-code"
➜  PCODE - 1 ./"p-code"
Please enter the instructions (Instruction END 0 0 to stopstack the input)
INT 0 4
LIT 0 100
STO 0 3
LOD 0 3
LOD 0 3
OPR 0 4
LOD 0 3
LIT 0 1
OPR 0 3
STO 0 3
LOD 0 3
JPC 0 17
LOD 0 3
LOD 0 3
OPR 0 4
OPR 0 2
JMP 0 6
STO 0 3
OPR 0 0
END 0 0
```

Compiled successfully!



```
LOD     0       3       1       11      0 0 0 1 338349 1
JPC     0       17      338349  12      0 0 0 1 338349
LOD     0       3       1       13      0 0 0 1 338349 1
LOD     0       3       1       14      0 0 0 1 338349 1 1
OPR     0       4       1       15      0 0 0 1 338349 1
OPR     0       2       338350  16      0 0 0 1 338350
JMP     0       6       338350  6       0 0 0 1 338350
LOD     0       3       1       7       0 0 0 1 338350 1
LIT     0       1       1       8       0 0 0 1 338350 1 1
OPR     0       3       0       9       0 0 0 1 338350 0
STO     0       3       338350  10      0 0 0 0 338350
LOD     0       3       0       11      0 0 0 0 338350 0
JPC     0       17      338350  17      0 0 0 0 338350
STO     0       3       338350  18      0 0 0 338350
OPR     0       0       0       0       0
➜  PCODE - 1
```

Exercício 05:

Soma dos cubos dos números naturais de 1 até 100(iterativamente);

Resolução:

```
➜  PCODE - 1 ./"p-code"
Please enter the instructions (Instruction END 0 0 to stopstack the input)
INT 0 4
LIT 0 100
STO 0 3
LOD 0 3
LOD 0 3
LOD 0 3
OPR 0 4
OPR 0 4
LOD 0 3
LIT 0 1
OPR 0 3
STO 0 3
LOD 0 3
JPC 0 21
LOD 0 3
LOD 0 3
LOD 0 3
OPR 0 4
OPR 0 4
OPR 0 2
JMP 0 8
STO 0 3
OPR 0 0
END 0 0_
```

```
LOD     0       3       1       13      0 0 0 1 25502499 1
JPC     0       21      25502499        14      0 0 0 1 25502499
LOD     0       3       1       15      0 0 0 1 25502499 1
LOD     0       3       1       16      0 0 0 1 25502499 1 1
LOD     0       3       1       17      0 0 0 1 25502499 1 1 1
OPR     0       4       1       18      0 0 0 1 25502499 1 1
OPR     0       4       1       19      0 0 0 1 25502499 1
OPR     0       2       25502500        20      0 0 0 1 25502500
JMP     0       8       25502500        8       0 0 0 1 25502500
LOD     0       3       1       9       0 0 0 1 25502500 1
LIT     0       1       1       10      0 0 0 1 25502500 1 1
OPR     0       3       0       11      0 0 0 1 25502500 0
STO     0       3       25502500        12      0 0 0 0 25502500
LOD     0       3       0       13      0 0 0 0 25502500 0
JPC     0       21      25502500        21      0 0 0 0 25502500
STO     0       3       25502500        22      0 0 0 25502500
OPR     0       0       0       0       0
➜  PCODE - 1
```

Exercício 06:

Main chama função Fat(5);

Resolução:

```
➜  PCODE - 1 ./"p-code"
Please enter the instructions (Instruction END 0 0 to stopstack the input)
INT 0 4
LIT 0 5
STO 0 3
LOD 0 3
STO 0 7
CAL 0 9
LOD 0 7
STO 0 3
OPR 0 0
INT 0 4
LOD 0 3
LOD 0 3
LIT 0 1
OPR 0 3
STO 0 3
LOD 0 3
JPC 0 20
LOD 0 3
OPR 0 4
JMP 0 11
STO 0 3
OPR 0 0
END 0 0_
```

Compiled successfully!

```
JPC     0     20     120     17     1 1 6 1 120
LOD     0     3      1       18     1 1 6 1 120 1
OPR     0     4      120     19     1 1 6 1 120
JMP     0     11     120     11     1 1 6 1 120
LOD     0     3      1       12     1 1 6 1 120 1
LIT     0     1      1       13     1 1 6 1 120 1 1
OPR     0     3      0       14     1 1 6 1 120 0
STO     0     3      120     15     1 1 6 0 120
LOD     0     3      0       16     1 1 6 0 120 0
JPC     0     20     120     20     1 1 6 0 120
STO     0     3      120     21     1 1 6 120
OPR     0     0      5       6      0 0 0 5
LOD     0     7      120     7      0 0 0 5 120
STO     0     3      120     8      0 0 0 120
OPR     0     0      0       0      0
➜  PCODE - 1 _
```

Exercício 07:

Main chama função Fib(4);

Extra: Fib(10);

Resolução:

```
➜  PCODE - 1 ./"p-code"
Please enter the instructions (Instruction END 0 0 to stopstack the input)
INT 0 4
LIT 0 4
STO 0 3
LOD 0 3
STO 0 7
CAL 0 9
LOD 0 7
STO 0 3
OPR 0 0
INT 0 6
LIT 0 1
LIT 0 0
STO 0 4
STO 0 5
LOD 0 3
JPC 0 27
LOD 0 5
LOD 0 4
LOD 0 5
OPR 0 2
STO 0 5
STO 0 4
LOD 0 3
LIT 0 1
OPR 0 3
STO 0 3
JMP 0 14
STO 0 3
OPR 0 0
END 0 0
```

```
STO     0       4       5       22      1 1 6 1 3 5
LOD     0       3       1       23      1 1 6 1 3 5 1
LIT     0       1       1       24      1 1 6 1 3 5 1 1
OPR     0       3       0       25      1 1 6 1 3 5 0
STO     0       3       5       26      1 1 6 0 3 5
JMP     0       14      5       14      1 1 6 0 3 5
LOD     0       3       0       15      1 1 6 0 3 5 0
JPC     0       27      5       27      1 1 6 0 3 5
STO     0       3       3       28      1 1 6 5 3
OPR     0       0       4       6       0 0 0 4
LOD     0       7       5       7       0 0 0 4 5
STO     0       3       5       8       0 0 0 5
OPR     0       0       0       0       0
➜  PCODE - 1
```

Fib(10):

```
LOD     0       3       0       15      1 1 6 0 55 89 0
JPC     0       27      89      27      1 1 6 0 55 89
STO     0       3       55      28      1 1 6 89 55
OPR     0       0       10      6       0 0 0 10
LOD     0       7       89      7       0 0 0 10 89
STO     0       3       89      8       0 0 0 89
OPR     0       0       0       0       0
➜  PCODE - 1
```

Exercício 08:

Main chama função Fat(5) Recursiva;

Resolução:

```
1   INT 0 4
2   LIT 0 5
3   STO 0 3
4   LOD 0 3
5   STO 0 7
6   CAL 0 8
7   LOD 0 7
8   OPR 0 0
9
10  INT 0 4
11  LOD 0 3
12  JPC 0 21 // SE FOR 0 PULA PRA
13  LOD 0 3
14  LIT 0 1
15  OPR 0 3 // O BUFFER ESTÁ COM n-1
16  STO 0 7 //  ENVIA n-1 COMO PARAMETRO
17  CAL 0 8 /   CHAMADA RECURSIVA
18  LOD 0 7 //  LOAD O VALOR RESULTADO DA RECURSÃO
19  LOD 0 3
20  OPR 0 4 // MULTIPLICA
21  STO 0 3 // ARMAZENA SER ACESSADO NO RETORNO
22  OPR 0 0
23
24  LIT 0 1
25  STO 0 3 // ARMAZENA 1 NO RETORNO CASO n == 0
26  OPR 0 0
```

```
LOD     0       3       4       19      5 5 17 4 6 4
OPR     0       4       24      20      5 5 17 4 24
STO     0       3       24      21      5 5 17 24
OPR     0       0       5       17      1 1 6 5
LOD     0       7       24      18      1 1 6 5 24
LOD     0       3       5       19      1 1 6 5 24 5
OPR     0       4       120     20      1 1 6 5 120
STO     0       3       120     21      1 1 6 120
OPR     0       0       5       6       0 0 0 5
LOD     0       7       120     7       0 0 0 5 120
STO     0       3       120     8       0 0 0 120
OPR     0       0       0       0       0
➜  PCODE - 1 _
```

Exercício 09:

Main chama função Fib(4) Recursiva;

Extra: Fib(10) Recursiva;

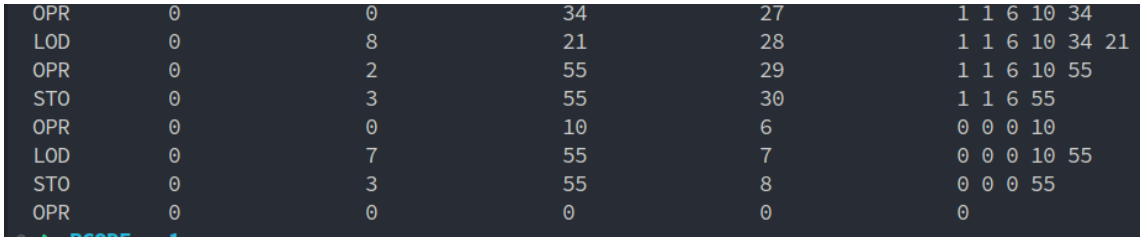
Resolução:

```
1  INT 0 4
2  LIT 0 4
3  STO 0 3
4  LOD 0 3
5  STO 0 7
6  CAL 0 9
7  LOD 0 7
8  STO 0 3
9  OPR 0 0
10
```

```
11  INT 0 4
12  LOD 0 3
13  JPC 0 30        // SE FOR 0 PULA PARA RETURN
14  LOD 0 3
15  LIT 0 1        //CASO FOR 1 ELE DIMINUI 1 E VERIFICA SE É 1 NA PROXIMA LINHA
16  OPR 0 3
17  JPC 0 30        // SE FOR 1 PULA PARA RETURN
18  LOD 0 3
19  LIT 0 1
20  OPR 0 3  // O BUFFER ESTÁ COM | n-1 |
21  STO 0 7     //ENVIA n-1 COMO PARAMETRO
22  CAL 0 9     //CHAMADA RECURSIVA
23  LOD 0 7     // LOAD O VALOR RESULTADO DA RECURSÃO
24  LOD 0 3
25  LIT 0 2
26  OPR 0 3  // O BUFFER ESTÁ COM  | fib(n-1) | n-2 |
27  STO 0 8     //ENVIA n-2 COMO PARAMETRO
28  CAL 0 9     //CHAMADA RECURSIVA
29  LOD 0 8     // LOAD O VALOR RESULTADO DA RECURSÃO
30  OPR 0 2     // SOMA O BUFFER ATUAL = | fib(n-1) | fib(n-2) |
31  STO 0 3     // ARMAZENA PARA O VALOR RETURN
32  OPR 0 0
33  END 0 0
```

```
STO       0         3         1         30        5 5 27 1
OPR       0         0         2         27        1 1 6 4 2
LOD       0         8         1         28        1 1 6 4 2 1
OPR       0         2         3         29        1 1 6 4 3
STO       0         3         3         30        1 1 6 3
OPR       0         0         4         6         0 0 0 4
LOD       0         7         3         7         0 0 0 4 3
STO       0         3         3         8         0 0 0 3
OPR       0         0         0         0         0
➜  PCODE - 1
```

Fib(10) Recursiva:



```
OPR     0       0       34      27      1 1 6 10 34
LOD     0       8       21      28      1 1 6 10 34 21
OPR     0       2       55      29      1 1 6 10 55
STO     0       3       55      30      1 1 6 55
OPR     0       0       10      6       0 0 0 10
LOD     0       7       55      7       0 0 0 10 55
STO     0       3       55      8       0 0 0 55
OPR     0       0       0       0       0
```

REFERÊNCIA

https://en.wikipedia.org/wiki/P-code_machine

https://trendspdf.prograd.uesc.br/MaterialApoio/Diario/Aula/2002681954/Sebesta_Cap10pp.pdf