

Diferença absoluta entre os números pares e ímpares na sequência inversa Fibonacci

Requested files: user.c, input.txt (Download)

Type of work: Individual work

Grade settings: Maximum grade: 10

Run: Yes **Evaluate:** Yes

Automatic grade: Yes

Definimos a sequência inversa Fibonacci de base n e m , denotada por " $\text{inv}(F_{n,m})$ " como a sequência de números inteiros positivos na qual cada termo sub-sequente corresponde à diferença dos dois números anteriores. Por exemplo, a sequência inversa Fibonacci de base 81 e 50 " $\text{inv}(F_{81,50})$ " é a sequência de números:

81, 50, 31, 19, 12, 7, 5, 2, 3

A sequência inversa Fibonacci é iniciada pelos números n e m , sendo n sempre maior do que m ($n > m$) e ela termina quando a diferença dos dois números anteriores n e m é menor ou igual do que 0. No exemplo acima, termina quando $2-3=-1$.

Escreva um programa usando a linguagem de programação C, que dados os valores de n e m , determine a diferença absoluta entre os números pares e ímpares na sequência inversa Fibonacci de base n e m . Por exemplo, para a sequência acima, o resultado do programa será $82 = (81+31+19+7+5+3)-(50+12+2)$.

Dicas:

- Para calcular o valor absoluto de um número x , uma solução básica é multiplicar o número por -1 se x é menor do que 0 (zero).

Entrada e Saída:

A entrada será constituída por pares de números inteiros n e m separados por um espaço em branco. Todos os inteiros serão números maiores do que 0, assim como sempre n será maior do que m . Cada linha no arquivo "input.txt" representará uma entrada para o programa.

Como saída você deverá imprimir três números, os valores de n , m e a diferença absoluta entre os números pares e ímpares na sequência inversa Fibonacci de base n e m (na sequência são incluídos os números n e m).

<u>Exemplos de entrada</u>	<u>Saída para os exemplos de entrada</u>
81 50	81 50 82
34 21	34 21 0
1189 360	1189 360 1658
832040 514229	832040 514229 0

Requested files

user.c

```

1 #include <stdio.h>
2
3 int main() {
4     int n, m;
5     scanf("%d %d", &n, &m);
6     // escreva seu código aqui
7     return 0;
8 }
9

```

input.txt

```

1 34 21
2 1189 360
3 832040 514229
4 34523412 31523412

```

Execution files

vpl_run.sh

```

1 #!/bin/bash
2 cat > vpl_execution <<EE00FF
3 #!/bin/bash
4 prog1="user"
5 prog2="test"
6 gcc \${prog1}.c -o \${prog1} -lm | grep -v Note > grepLines.out
7 gcc \${prog2}.c -o \${prog2} -lm | grep -v Note > grepLines.out
8 if [ -s grepLines.out ] ; then
9     echo "ERROS no compilador"
10    cat grepLines.out
11    exit
12 fi
13
14 while IFS='' read -r line || [[ -n "\${line}" ]]; do
15     if [[ ! -z "\${line}" ]]; then
16         echo "\${line}" > in.txt
17         echo -e "-----Saída de seu programa-----"
18         echo -e "Para a entrada: \${line}"
19         echo -e "-----"
20         ./user < in.txt
21         echo -e "\n"
22
23         echo -e "-----Saída esperada do programa-----"
24         echo -e "Para a entrada: \${line}"
25         echo -e "-----"
26         ./test < in.txt
27         echo -e "\n"
28     fi
29 done < input.txt
30 EE00FF
31
32 chmod +x vpl_execution
33

```

vpl_debug.sh

vpl_evaluate.sh

```

1 #!/bin/bash
2 cat >vpl_execution << 'EOF'
3 #!/bin/bash
4
5     user="user"
6     test="test"
7     params_file="params.in"
8     input_tests="tests.in"
9
10    # > Compile the executable
11    gcc $user.c -o $user -lm
12    gcc $test.c -o $test -lm
13
14    # > The params file format:
15    # .. First line: number of tests;
16    # .. Second line: number of inputs from each test
17    typeset -i num_tests=$(head -n 1 $params_file)
18    typeset -i num_input=$(tail -n 1 $params_file)
19    declare -i successes=0
20
21    # > Read every test ...
22    for num in $(seq 0 $((num_tests-1)));
23    do
24        > "in.txt"
25        # ... param by param, composing an input file
26        for input in $(seq 0 $((num_input-1)));
27        do
28            typeset -i selected_line=$((num*num_input)+input+1))
29            cmd="$selected_line!d"
30            sed $cmd $input_tests >> "in.txt" # get the selected input from file
31        done
32
33        # > Execute both user and test programs with the same input
34        echo `./$user < in.txt` > ${user}_out
35        echo `./$test < in.txt` > ${test}_out
36
37        diff -y -w -B --ignore-all-space ${user}_out ${test}_out > diff.out
38        # > Wrong answer
39        if (($? > 0)); then
40            echo "Comment :=>> Incorrect output found on test $num"
41            echo "Comment :=>>- Your output"
42            echo "<|--"
43            cat ${user}_out
44            echo "--|>"
45            echo ""
46            echo "Comment :=>>- Expected output "
47            echo "<|--"
48            cat ${test}_out
49            echo "--|>"
50
51            # > Right answer
52        else
53            successes=$((successes+1))
54            #echo "Comment :=>> Correct output."
55        fi
56    done
57    echo "-----"
58    echo "Comment :=>>- Your success rate is ${successes}/${num_tests}."
59    echo "Grade :=>> $((10*successes)/num_tests))"
60    echo "-----"
61
62 EOF
63
64 chmod +x vpl_execution
65

```

vpl_evaluate.cases

test.c

```
1 #include <stdio.h>
2
3 int main() {
4     int n, m, N, M, accum = 0;
5     scanf("%d %d", &N, &M);
6     n = N; m = M;
7     if (n%2==0) accum+=n; else accum-=n;
8     if (m%2==0) accum+=m; else accum-=m;
9     while (n-m>0) {
10         int m_temp = m;
11         m = n - m;
12         n = m_temp;
13         if (m%2==0) accum+=m; else accum-=m;
14     }
15     if (accum<0) accum *= -1;
16     printf("%d %d %d\n", N, M, accum);
17     return 0;
18 }
19
```

params.in

```
1 10
2 2
```

tests.in

```
1 34
2 21
3 1189
4 360
5 832040
6 514229
7 34523412
8 31523412
9 514229
10 28657
11 2189
12 660
13 54553312
14 21583412
15 614229
16 38657
17 2380
18 760
19 23890
20 6770
```

