Distância dos rebates da bola de elástico

Requested files: user.c, input.txt (Download)
Type of work: Individual work
Grade settings: Maximum grade: 10 **Hidden**
Run: Yes **Evaluate**: Yes **Evaluate just on submission**: Yes
Automatic grade: Yes

Quando uma bola de elástico é jogada de uma altura de n metros, ela rebate até dois terços da altura *n* mais 1 metro, se *n* é divisível por 3. Se a altura *n* da qual cai é par (divisível por 2), ela rebate até a metade da altura *n* mais 1. Caso contrário ela rebate até uma altura que é a soma dos divisores próprios de *n*. O processo é repetido com o novo valor da altura *n* alcançada pelo último rebate, sendo que o processo termina no momento em que *n* é menor do que 10 metros ($n < 10$).

Escreva um algoritmo que, dados um número *n*, calcule a distância total percorrida pelos rebates da bola de elástico, Por exemplo, se a seguinte sequência 447, 299, 37, 1 corresponde à sequência de rebates que é gerada para a caída da altura *n*=669, a distância total de repiques será 784=447+299+37+1.

Dicas:

- O operador de resto em C é %, assim para calcular o resto de um número *x* entre 2 na variável resto deve ser escrita a linha:
  `resto = x % 2;`
- Divisores próprios de um número positivo *n* são todos os divisores inteiros positivos de *n* exceto o próprio *n*. Por exemplo, os divisores próprios do número 30 são 1, 2, 3, 5, 6, 10 e 15.

Entrada e Saída:

A entrada será constituída por um número inteiro *n* que representam a altura inicial da caída da bola de elástico, sendo que *n* é sempre maior que 10. Cada linha no arquivo "input.txt" representará uma entrada para o programa. Como saída você deve imprimir dois números separados por um espaço em branco, os quais são: o valor de *n* e a distância percorrida pelos repiques da bola de elástico.

Exemplos de entrada	**Saída para os exemplos de entrada**
669	669 784
750	750 693
1043365513	1043365513 1924216

Requested files

user.c

```c
#include <stdio.h>

int main() {
          int n;
          scanf("%d",&n);
          // escreva seu código aqui

          return 0;
}
```

input.txt

```
669
750
1043365513
```

Execution files

vpl_run.sh

```
#! /bin/bash

cat > vpl_execution <<EEOOFF
#! /bin/bash
prog1="user"
prog2="test"
gcc \${prog1}.c -o \${prog1} -lm | grep -v Note > grepLines.out
gcc \${prog2}.c -o \${prog2} -lm | grep -v Note > grepLines.out
if [ -s grepLines.out ] ; then
     echo "Some compiler ERRORS reported"
     cat grepLines.out
     exit
fi

while IFS='' read -r line || [[ -n "\${line}" ]]; do
    if [[ !  -z  "\${line}"  ]]; then
        echo "\${line}" > in.txt
        echo "----------Your program----------"
        ./\${prog1} < in.txt
        echo "--------------------"
        echo "----------Test program----------"
        ./\${prog2} < in.txt
        echo "--------------------"
    fi
done < input.txt
EEOOFF

chmod +x vpl_execution
```

vpl_debug.sh

```

```

vpl_evaluate.sh

```
#!/bin/bash

cat >vpl_execution << 'EOF'
#!/bin/bash
user="user"
test="test"
params_file="params.in"

# > Compile the executable
gcc $user.c -o $user -lm
gcc $test.c -o $test -lm

# > The params file format:
# .. First line: number of tests;
# .. Second line: number of inputs from each test
typeset -i num_tests=$(head -n 1 $params_file)
typeset -i num_input=$(tail -n 1 $params_file)
declare -i successes=0

# > Read every test ...
for num in $(seq $num_input $(($num_tests+$num_input-1)));
do
    echo $num
    echo $num > "in.txt"
    # > Execute both user and test programs with the same input
    echo `./$user < in.txt` > ${user}_out
    echo `./$test < in.txt` > ${test}_out

    diff -y -w -B --ignore-all-space ${user}_out ${test}_out > diff.out
    # > Wrong answer
    if (($? > 0)); then
        echo "Comment :=>> Incorrect output found on n = $num"
        echo "Comment :=>>- Your output"
        echo "<|--"
        cat ${user}_out
        echo "--|>"
        echo ""
        echo "Comment :=>>- Expected output "
        echo "<|--"
        cat ${test}_out
        echo "--|>"
    # > Right answer
    else
        successes=$((successes+1))
        #echo "Comment :=>> Correct output."
    fi
done
echo "------------------------------"
echo "Comment :=>>- Your success rate is ${successes}/${num_tests}."
echo "Grade :=>>$(((10*successes)/num_tests))"
echo "------------------------------"

EOF

chmod +x vpl_execution
```

vpl_evaluate.cases

test.c

```c
#include <stdio.h>

int main() {
    int n;
    scanf("%d", &n);

          int altura = n, accum = 0;

          while (altura >= 10) {
        if (altura % 3 == 0) {
                  altura = (2 * altura / 3) + 1;
          } else if (altura % 2 == 0) {
            altura = (altura / 2) + 1;
          } else {
                  int divisor, sum = 1;
                  for (divisor = 2 ; divisor*divisor <= altura; divisor++) {
                  if (altura % divisor == 0) {
                      sum = sum + divisor;
                      if (altura / divisor != divisor) sum = sum + (altura/divisor);
                  }
                  }
                  altura = sum;
          }
          accum = accum + altura;
          }
          printf("%d %d\n", n, accum);
          return 0;
}
```

params.in

```
1000
11
```

VPL 3.1.4
