**1º Trabalho Prático**

**Programação**

Realizado por:

<46055> João Martins

<46001> José Santos

<46074> Ricardo Margalhau

Programa – Number

* No programa Number começámos por analisar o input inserido pelo o utilizador na consola através do Scanner (java.util.Scanner). Em seguida para obter os milhares foi dividido por 1000 conservando-o numa variável do tipo Inteiro. Posteriormente para encontrar o dígito correspondente à casa das centenas dividimos o número original por 100 subtraindo-lhe o valor dos milhares multiplicado por 10, por exemplo, considerando o número 2100 temos que os milhares são (int) (2100 / 1000) = 2 e as centenas (int) (2100 / 100 – 2 \* 10) = 1. Repetindo este processo para as seguintes casas do número original podemos obter os milhares, as centenas, as dezenas e as unidades.

Programa – Slope

* Para realizar o programa Slope começámos por obter os 4 inputs vindos do utilizador (x1, y1, x2, y2) e verificar se x1 e x2 são iguais e y1 e y2 são iguais. Neste caso o programa iria encontrar uma indeterminação (0 / 0) pelo que enviamos uma mensagem de erro ao utilizador caso isto se venha a verificar.
* Depois verificámos se o x1 é igual ao x2 e caso isto venha acontecer a reta não apresenta declive pois irá tratar-se de uma reta do tipo y = b
* Caso ambas as opções acima sejam falsas então temos o declive pela fórmula matemática da reta e substituímos um ponto para encontrar a ordenada na origem (b).

Programa – Dice

* Em Dice iniciámos o programa calculando o número do primeiro dado através do método *random* da class *Math* (java.lang.Math). Nós conseguimos obter um número random entre 1 e 6 inclusive através da seguinte manipulação matemática:

0 <= random < 1

0 \* 6 <= random \* 6 < 1 \* 6

0 + 1 <= random \* 6 + 1 < 6 + 1

1 <= random \* 6 + 1 < 7

1 <= (int) (random \* 6 + 1) <= 6

* Após o utilizador premir a tecla *Enter* no seu teclado (verificámos isto com o método *nextLine* da class Scanner) calculámos o segundo número da mesma forma que o primeiro. Em seguida utilizámos uma operação ternária dentro da instrução *println* para verificar se os dois números gerados são iguais. Caso a comparação venha a ser verdade temos que calcular o total da soma a dobrar, caso a comparação não se verifique apenas apresentamos o total da soma destes dois números.

Programa – Count4

* No programa Count4 para obtermos os números ímpares podemos somar o resto das divisões de cada número inserido por 2, pois cada número ímpar tem resto 1 na divisão por 2. Para o cálculo dos números pares basta subtrair o número total de ímpares, realizado anteriormente, a 4 (número total).
* A próxima etapa deste programa foi calcular as igualdades entre números, sendo que para isto começámos por analisar se todos eles são iguais e, caso não sejam, fomos testar a igualdade entre 3 e 2 números sendo que se nenhum for igual temos que o número de números iguais é zero.

Programa – Travel

* Para o programa Travel começámos por ler os 6 inputs pedidos ao utilizador final (hora de partida, minuto da partida, segundo da partida, hora da chegada, minuto da chegada e segundo da chegada) e por fazer diferenças entre a chegada e a partida da seguinte forma:

Hora chegada – Hora partida

Minuto chegada – Minuto partida

Segundo chegada – Segundo partida

* Após estas diferenças calculadas fomos analisar se o tempo da partida é após o da chegada. Podemos analisar esta parte através do seguinte:

Se diferença das horas < 0, então, partida > chegada

ou

Se diferença das horas == 0 e se a diferença dos minutos < 0, então partida > chegada

ou

Se diferença das horas == 0 e se a diferença dos minutos == 0 e se a diferença dos segundos < 0, então, partida > chegada

* Caso a partida <= chegada temos que se a diferença dos minutos for menor que zero então devemos somar 60 aos minutos e subtrair 1 à hora devido a soma que efetuámos aos minutos. Isto repete-se para os segundos, visto que, caso a diferença dos segundos seja < 0, somamos 60 aos segundos e subtraímos 1 dos minutos.