Feedback Lista1 LP

Raryson Fred da Silva Gomes - 20200138154

Este documento tem por objetivo informar relatos acerca da primeira lista de exercícios da disciplina de Linguagem de Programação I (LPI). Assim, apresentando de forma objetiva e direta os dados obtidos na resolução da atividade. Como o valor total da atividade equivale a 02 pontos e, levando em consideração as 08 questões, têm-se:

$$x = \frac{8}{2} = 0{,}25$$

Com isso, segue abaixo a tabela contendo todas as questões, possuindo suas respectivas notas.

Questão	Acerto	SubTotal	Peso Questão
fibonacci	90%	0,225	0,25
intervalos	95%	0,2375	
inverter	85%	0,2125	
minmax	95%	0,2375	
negativos5	95%	0,2375	
ponto_em_retangulo_1	70%	0,175	
ponto_em_retangulo_2	60%	0,15	
soma_vizinhos	90%	0,225	
	Total	1,7	

Logo, ao analisar, de forma resumida, pode-se elencar os itens contidos acima, fazendo observações individualmente.

Fibonacci:

Não houve problemas no entendimento da proposta de questão, mas apenas no modo de apresentar as informações, tal como esperava-se na saída do programa, definindo a sequência dentro de colchetes. Além disso, inicialmente obtive dificuldades em utilizar o *push_back*.

```
9       val.push_back(x);
10      val.push_back(y);
```

Intervalos:

Não houve problemas no entendimento da proposta da questão e em nenhum outro fator, mas a maior dificuldade se deu na verificação de onde partir e finalizar o intervalo. Houve também dificuldades em utilizar o *setprecision*, tendo em vista as regras ligeiramente distintas de outras linguagens de programação.

```
6       using std::setprecision;
```

Porém, como em outras linguagens a formatação de controle de casas decimais se dá com uma restrição na *String*, tal como em Java. Assim, apresentando-se como:

```
1     System.out.println("%.2f", (variavelQualquer));
```

Por outro lado, em C++, têm-se a necessidade de passar como parâmetro um inteiro *int* variavelQualquer = 4 para formatar um valor.

Inverter:

Houve problemas no entendimento da proposta da questão. Então, apesar de ser uma proposta relativamente simples, não foi possível executar como na correção da questão.

Minmax:

Não houve problemas no entendimento da proposta da questão. No entanto, na classe *pair<int, int>* não ficou claro, bem como o *make_pair*, fornecido na correção do exercício.

```
25    return std::make_pair(min, max);
```

Negativos5:

Não houve problemas no entendimento da proposta da questão e em nenhum outro fator, uma vez que se configurou como a mais fácil dentre as oito propostas. No entanto, ocorreu um determinado erro ao utilizar o trecho de código abaixo, uma vez que deveria usar a constante inteira já definida antes da função *main*.

```
8     int vetor[SIZE];
```

Código fornecido anteriormente:

```
5     const int SIZE = 5; // input size.
```

Ponto_em_retangulo_1:

Houve poucos problemas no entendimento da proposta da questão. Assim, por requisitar um raciocínio elevado, algumas ideias não foram bem executadas. De forma geral, a construção do algoritmo não se relacionou totalmente com a resposta da lista, sendo preciso apenas o seguinte trecho de código:

```
11    location_t loc = location_t::BORDER;
12    if(P.x < IE.x || P.x > SD.x || P.y < IE.y || P.y > SD.y){
13          loc = location_t::OUTSIDE;
14    } else if(P.x > IE.x && P.x < SD.x && P.y > IE.y && P.y < SD.y){
15          loc = location_t::INSIDE;
16    }
17    return loc;
```

Ponto_em_retangulo_2:

Houve maiores problemas no entendimento da proposta da questão, também de forma análoga à questão anterior.

Soma_vizinhos:

Não houve problemas no entendimento da proposta da questão e em nenhum outro fator. Porém, de modo geral, não se deveria utilizar esse tipo de condição para uma estrutura de repetição infinitos. Logo, poderia passar as informações nesse mesmo laço para definir um intervalo de repetição, tal como uma variável para controlar onde iniciará e outra definindo o fim da soma.

```
10    while(1){
            .
            .
            .
33    }
```