Aula 01 – Exercícios

Parte 1 – Preliminares

1) Criar uma conta no site: https://run.codes/

Esta ferramenta será usada para a avaliação dos códigos de programação que forem desenvolvidos durante o decorrer da disciplina. No cadastro, use dados reais (nome, matrícula e um email sério). Caso contrário, o professor não consegue te identificar na hora de passar a sua nota.

2) Após criar a conta, confirmar o email e fazer login no Run Codes, insira no campo "Código de Matrícula" a seguinte chave de acesso: **6KY2**

Na seção "Minhas Disciplinas", clique no botão "Ver Página da Disciplina" para acessar a área da disciplina de Algoritmos e Estruturas de Dados 2, onde estarão disponíveis os Exercícios que devem ser entregues.

Para entregar um exercício, na linha da tabela correspondente a ele, clique em "Ver Detalhes". Na tela que se abre, haverá um botão vermelho chamado "Selecionar Arquivo". É através dele que você enviará um código fonte com extensão **.c** para ser avaliado. Após selecionar o arquivo, deve-se confirmar o envio. O Run Codes irá testar o código enviado e atribuir uma nota de 0 a 10 a cada exercício entregue. Você pode realizar quantas submissões desejar, sendo que a maior nota obtida será considerada.

Após o encerramento do prazo de entrega, o sistema também irá verificar por plágios nas submissões, mas não vai te notificar e nem te apresentar um desconto de nota.

3) A terceira instrução para a programação dos algoritmos é seguir ***EXATAMENTE*** os padrões de entrada e saída descritos nos enunciados.

Por exemplo, suponha o seguinte enunciado: "fazer um programa em C que leia uma quantidade N e, a seguir, leia N números e os armazene em um vetor.". O que deverá ser feito na função principal é:

```
1. int main()
2. {
3.     int N, i, vet[max];
4.
5.     scanf("%d", &N);
6.
7.     for(i = 0; i < N; i++)
8.         scanf("%d", &vet[i]);
9.
10.    // Continua...
```

No exemplo acima, note que não há interação com o usuário nas linhas 5 e 8. O programa não contém uma instrução do tipo `printf("Digite o valor de N: ");` ou `printf("Digite um elemento do vetor: ");` e existe uma boa razão para isso. Como o seu código será avaliado pela ferramenta Run Codes, você, programador, não vai "conversar" com o Run Codes durante a execução do código.

Considere que o programa abre sem exibir nenhuma mensagem e, no caso, um usuário já abriria o seu programa sabendo o que deve ser digitado como entrada e em qual ordem esses valores devem vir, sem a necessidade de mensagens informativas. É exatamente isso que o Run Codes vai fazer: inserir os valores dos casos de teste em seu programa e verificar se a resposta está correta. Tudo que você imprimir na tela usando `printf`, será considerado como parte da resposta. Atente-se, também, para a ordem em que os dados de entrada e saída estão descritos nos enunciados.

Parte 2 – Escrita (entregue via Formulário no Google Classroom)

Nos slides da Aula 01, foram deixados 4 exercícios para serem executados no papel com os algoritmos estudados. Em seu caderno, simule a execução dos exercícios correspondentes. Ao final, você deve escanear ou fotografar as páginas com a câmera do celular, e submetê-las no formulário que acompanha esta atividade. As páginas devem ser reunidas em um único arquivo PDF e devem estar legíveis. Atenção para enviar seu arquivo no Formulário disponível na atividade, e não na seção de upload do Classroom. Lembre-se de colocar nome nas suas folhas.

Parte 3 – Codificação (entregue via Run Codes)

Exercício 1: escreva, em linguagem C pura (sem nenhum comando próprio de C++), um programa que leia um valor N. Após criar um vetor de N posições, repasse-o, juntamente com o valor de N a uma função que leia e armazene valores no vetor. Por fim, execute o algoritmo Insertion Sort para ordenar o vetor e mostre, como resposta, uma única linha contendo dois valores numéricos, separados por um único espaço. O primeiro valor é a quantidade de comparações executadas pelo algoritmo de ordenação. O segundo valor é a quantidade de trocas realizadas.

Seu programa deve ser dividido em funções. É obrigatório que você escreva, pelo menos, as seguintes funções, seguindo os protótipos dados:

```
// esta função lê n elementos e os armazena no vetor v. Recebe como parâmetros o vetor v e a quantidade n.
void leVetor(int *v, int n);

// esta função ordena o vetor. Recebe como parâmetros o vetor v e o tamanho n do vetor.
void insertionSort(int *v, int n);
```

Exercício 2: repetir o Exercício 1, porém, para o algoritmo Selection Sort.

Observação: este exercício está dividido em uma parte escrita e uma parte prática de codificação. A entrega só será válida se as duas partes forem entregues.