Disciplina	**Data de Entrega das Soluções**
IAL-002 – Algoritmos e Lógica de Programação	13/07/2020

Nome dos Projetos Programa contidos neste documento

1. Cálculo de Médias
2. Totalização Simples de Vendas de Produtos
3. Gerador de Senhas
4. Gerador de dados de vendas – Este é o Desafio!!!

Requisitos para entrega

1. Atividade em grupo de 3 a 5 alunos. Entregas individuais podem, eventualmente, ser aceitas com autorização prévia do professor (para os casos de alunos que não sejam da turma).
2. Os programas devem ser entregues através da tarefa do Teams "Atividade de Avaliação N2 – Entrega em 13/07/2020".
3. É necessário que apenas um integrante do grupo faça o upload das soluções no Teams.
4. Cada projeto programa deve estar em um arquivo e os nomes dos arquivos estão indicados em vermelho no título do projeto.
5. O código fonte deverá conter no seu início o nome completo dos integrantes do grupo.
6. Os programas devem ser escritos em Linguagem Python 3. Podem usar as funções de listas que desejarem. Bibliotecas externas não podem ser usadas, com exceção da biblioteca *random*.

Projeto Programa 1 – Cálculo de Médias – arquivo pjp01.py

Descrição do Projeto

Informações Iniciais

Este projeto estará baseado na leitura de um arquivo texto de entrada contendo dados de alunos de uma disciplina da faculdade. O conteúdo deste arquivo estará formatado como CSV (Comma Separated Values).
Ele terá o nome **ALUNOS.TXT** e conterá dados conforme o layout mostrado abaixo. Nos campos numéricos reais lembre-se de usar o caractere ponto (.) como separador decimal.:

```
181312202;7.3;8.8;5.9;Ana Maria de Oliveira
181312198;4.4;3.7;7.6;Carlos Alberto Mancini
181212155;7.9;5.3;6.1;Eduardo Mendes Junqueira
...
```

Como pode ser visto acima, cada linha do arquivo refere-se a um aluno. Para os testes do programa deve-se usar um arquivo com pelo menos 25 alunos, ou seja, este arquivo terá pelo menos 25 linhas e as informações de cada linha estão descritas na tabela a seguir. É necessário que **vocês criem** esse arquivo usando um editor de texto simples como o bloco de notas, por exemplo.

Posição	Informação	Formato	Observações
(1)	Nº de matrícula do aluno	9 dígitos numéricos	
(2)	Nota P1	Número Real	Nota da primeira prova
(3)	Nota P2	Número Real	Nota da segunda prova
(4)	Nota MT	Número Real	Nota média dos trabalhos entregues
(5)	Nome do aluno	Texto	Este campo terá tamanho variável, contendo quantos caracteres forem necessários para cada nome de aluno

Pede-se neste Projeto Programa

O resultado que se espera deste programa é a exibição em tela da situação de cada aluno no padrão mostrado abaixo

```
                                  Média
Matrícula  Nome do Aluno           Final  Situação
181312202  Ana Maria de Oliveira     7.6  Aprovado
181312198  Carlos Alberto Mancini    4.8  Reprovado
181212155  Eduardo Mendes Junqueira  6.5  Aprovado
```

O cálculo da média final é feito conforme a fórmula: $MF = \frac{4.P1+4.P2+2.MT}{10}$

E o critério para aprovação é que a média final seja maior ou igual a 6.0

Projeto Programa 2 – Totalização simples de vendas de produtos – arquivo pjp02.py
Descrição do Projeto

Informações Iniciais

Este projeto estará baseado na leitura de um arquivo texto de entrada contendo código, quantidade e valor unitário de produtos vendidos. O conteúdo deste arquivo estará formatado como CSV (Comma Separated Values).
Ele terá o nome **VENDAS.TXT** e conterá dados conforme o layout mostrado abaixo. Nos campos numéricos reais lembre-se de usar o caractere ponto (.) como separador decimal.:

```
12455;14;17.90
11590;130;4.50
18300;12;16.43
12455;96;17.43
14570;50;6.12
...
```

Como pode ser visto acima, cada linha do arquivo refere-se a um produto vendido. Para os testes do programa utilize o arquivo de dados fornecido pelo professor juntamente com este enunciado. As informações de cada linha do arquivo estão descritas na tabela:

Posição	Informação	Formato	Observações
(1)	Código do produto	5 dígitos numéricos	Os códigos de produtos serão números entre 10000 e 21000
(2)	Quantidade	Número inteiro	
(3)	Preço unitário	Número Real	

Pede-se neste Projeto Programa

- O programa deve ler os dados do arquivo de entrada e carregá-los em memória usando listas do Python 3. Procurem pensar em uma forma esperta de fazer isso e descrevam através de comentários no código do programa o que elaboraram.
- Após a leitura o programa deve calcular e mostrar na tela o total geral vendido (somatória de todas as Quantidades x Preço Unitário).
- Em seguida deve permanecer em laço enquanto não for digitado zero (0). Para cada repetição do laço deve-se ler um código de produto, obrigatoriamente no intervalo [10000 e 21000] e mostrar na tela o total vendido para aquele produto.

```
Digite o código: 14330
Total vendido do produto 14330 = R$ 7326.14

Digite o código: 11200
Total vendido do produto 11200 = R$ 0.00

Digite o código: 55000
55000 Código inválido (deve ser entre 10000 e 21000)

Digite o código: 14500
Total vendido do produto 14500 = R$ 712.40

Digite o código: 0
Fim do programa
```

Projeto Programa 3 – Gerador de Senhas – arquivo pjp03.py
Descrição do Projeto

Este projeto estará baseado na leitura de um arquivo texto de entrada contendo números de matrícula de alunos de uma escola. Sabe-se que cada nº de matrícula tem 6 caracteres. O arquivo de entrada deve ter o nome MATR.TXT e para realizar os testes do programa vocês devem criá-lo usando um editor de texto simples (bloco de notas, por exemplo). Para cada nº de matrícula presente no arquivo deve ser gerada uma senha conforme as condições especificadas abaixo. Tanto o nº de matrícula como a senha gerada devem ser gravados no arquivo de saída SENHAS.TXT com o formato a seguir:
Exemplo: neste exemplo foram geradas senhas do tipo Alfanumérica 1, com 7 caracteres

MATR.TXT	**SENHAS.TXT**
330019	330019;318A89P;
414061	414061;E87H14M;
109229	109229;019MKX9;
827392	827392;313G093;
etc...	

Condições para geração de senhas
No início do programa, antes de efetuar a leitura do arquivo de entrada, o programa deve pedir que o usuário informe:

1. O tipo de senha:
    a. Numérica – conterá apenas algarismos;
    b. Alfabética – conterá apenas letras maiúsculas e minúsculas;
    c. Alfanumérica 1 – conterá letras maiúsculas e algarismos;
    d. Alfanumérica 2 – conterá letras maiúsculas, minúsculas e algarismos;
    e. Geral – conterá letras maiúsculas, minúsculas, algarismos e os caracteres ASCII [33, 46] e [58, 64]
2. O tamanho da senha – quantidade de caracteres que a mesma deve conter

Requisito importante que fará parte da avaliação

A senha deve ser gerada dentro de uma função denominada GeraSenha. Essa função receberá dois parâmetros, conforme a escolha do usuário para as duas informações descritas acima:

- Tipo: que é um caractere que deverá ser ‘a’, ‘b’, ‘c’, ‘d’, ‘e’ e definirá o tipo de senha gerada
- Tam: que é o tamanho da senha, um número inteiro

O retorno da função é um string contendo a senha gerada. Assim, o cabeçalho e o retorno da função devem ser:

```
def GeraSenha(Tipo, Tam):
    # aqui virá todo o código necessário para gerar a senha
    return senha
```

Tabela ASCII

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[ENG OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

Projeto Programa 4 – Gerador de dados de vendas – arquivo pjp04.py
Este é o desafio para quem desejar atingir nota 10,0 !!!
Descrição do Projeto

Informações Iniciais

Este projeto estará baseado na leitura de um arquivo texto de entrada contendo dados básicos de produtos comercializados por uma empresa. Este arquivo estará formatado como CSV (Comma Separated Values).
Ele terá o nome **PRODUTOS.TXT** e conterá dados conforme o layout mostrado abaixo. Nos campos numéricos reais lembre-se de usar o caractere ponto (.) como separador decimal.

```
11370;P;14.352;17.35;18.82;
19258;U;317.000;17.80;19.30;
20412;U;19.000;4.75;27.55;
32177;P;16.120;5.38;23.00;
etc...
```

Como pode ser visto acima, cada linha do arquivo refere-se a um produto cadastrado que contém cinco informações separadas pelo caractere "ponto e vírgula" (;). Sugere-se que se trabalhe com 15 produtos distintos, ou seja, este arquivo terá 15 linhas e as informações de cada linha estão descritas na tabela abaixo:

Posição	Informação	Formato	Observações
(1)	Código do Produto	5 dígitos numéricos	
(2)	Modo de controle do estoque	P para peso e U para peças	Para os produtos vendidos a peso a unidade de medida é o Kg e o estoque é um número real com 3 casas decimais (gramas)
(3)	Quantidade em estoque no início do período	Número Inteiro ou Real	Este valor será lido como número real, mas nos casos de produtos vendidos por peça a parte decimal estará zerada 1.000
(4)	Preço de custo unitário do produto	Número Real	
(5)	Margem de lucro na venda	Número Real	Representa a porcentagem de margem de lucro que é aplicada na venda do produto

Pede-se neste Projeto Programa

O resultado que se espera deste programa é a produção de um arquivo de saída contendo dados de vendas com o layout exemplificado e descrito a seguir.

```
2016;11;01;11370;4.250;20.62;
2016;11;01;19258;5.000;21.23;
2016;11;01;19258;1.000;21.23;
2016;11;01;20412;19.000;1;6.05;
...
2016;11;02;32177;0.550;6.62;
2016;11;02;11370;1.780;19.80;
etc...
```

Posição	Informação	Formato	Observações
(1)	Ano da venda	4 dígitos	
(2)	Mês da venda	2 dígitos	Para meses de 1 a 9 o formato será 01 a 09
(3)	Dia da venda	2 dígitos	Para dias de 1 a 9 o formato será 01 a 09
(4)	Código do Produto	5 dígitos	
(5)	Quantidade vendida	Número Inteiro ou Real	Produtos vendidos a peso gravar com três casas decimais Produtos vendidos por peça gravar com três zeros decimais
(6)	Preço de venda	Número Real	O preço de venda é obtido aplicando-se a margem ao preço de custo.

Detalhes Adicionais

Período de vendas contido no arquivo

O período será o de um mês de vendas no arquivo. Esse é um dado de entrada. O usuário deverá digitar o mês e o ano desejado. O programa não deve aceitar meses inválidos. E o ano deve ser no mínimo 2016.

Quantidade de vendas a serem geradas

Acima nada se falou sobre a quantidade de vendas a ser gerada. No seu início o programa deverá ler do teclado a quantidade diária de lançamentos de vendas. Esta quantidade será fornecida pelo usuário e carregada em uma variável QtdeVendasDia.

O arquivo deverá ter o número de linhas QtdeVendasDia * NDiasMes, onde NDiasMes é a quantidade de dias do mês para o qual se deseja gerar as vendas.

Valores de venda diferenciados

O valor unitário de venda é obtido aplicando-se a margem ao preço de custo. Em alguns casos, porém, pode haver variações para menos ou para mais nesse preço de venda. Clientes maus pagadores são sobretaxados enquanto que compradores fiéis e frequentes acabam recebendo descontos.

Assim, pede-se que em 35% dos casos os lançamentos de vendas tenham valores de venda diferenciados, oscilando entre -8% e +8%

Requisitos Bônus

Sugere-se a quem desejar cumprir os requisitos bônus que entreguem duas versões deste projeto-programa. A primeira com os requisitos básicos e a segunda com os bônus. Assim se a versão com bônus não funcionar direito vocês garantirão uma nota melhor com a versão básica.

1. Faça com que o programa seja capaz de gerar lançamentos de vendas para todos os dias existentes em um certo período fornecido pelo usuário. O usuário digitará 6 números inteiros, sendo Dia, Mês e Ano de início do período e Dia, Mês e Ano do término do período. Este requisito bônus substitui a entrada de dados padrão solicitada acima, por uma entrada com mais dados.

2. A empresa não abre aos domingos. Assim, ao gerar o arquivo de saída não devem ser geradas vendas para dias de domingo. Para saber se um determinado dia é domingo use a função `DiaSemana` abaixo.
   Esta função recebe três inteiros: dia (dd), mês (mm) e ano (aa). Ela retorna 0 para segunda-feira, 1 para terça-feira, 2 para quart-feira e assim por diante. Para domingo ela retornará o valor 6. É necessário incluir no início do programa a importação da função date da biblioteca datetime.
   ATENÇÃO: Esta é uma exceção à regra de uso de bibliotecas externas. Quem for fazer este item pode usar a biblioteca datetime indicada a seguir

```
from datetime import date

def DiaSemana(int dd, int mm, int aa) {
    d = date(aa, mm, dd)
    return d.weekday()
}
```

3. Para cada venda agregue mais duas informações: a Unidade da Federação (Estado) para onde foi feita a venda e a alíquota de ICMS desta venda. Use a sigla dos Estados. As alíquotas de ICMS para as várias UFs estão na tabela ao lado e, embora sejam uma porcentagem, podem ser tratadas como números inteiros.

Unidade da Federação	Alíquota de ICMS
São Paulo	18%
Sudeste e Sul	12%
Zona Franca de Manaus	0%
Demais UFs	7%

   Sugestão: crie um struct com dois campos: Sigla da UF e a Aliq. ICMS. Crie um vetor desses structs e carregue com as siglas e alíquotas de todos a UFs. Ao gerar um lançamento de vendas randomize um número inteiro e use-o como indexador desse vetor.