Computação I – Enunciado do Trabalho

O aluno deve entrar em contato com a professora até o dia 28/01/2021 para falar o tema que pretende desenvolver.

O aluno deve entrar em contato com a professora até o dia 26/02/2021 para marcar a data da apresentação ou informar se irá gravar o vídeo.

Data de entrega máxima: semana do dia 1 de março de 2021 (última semana do período).

Para o trabalho da disciplina de Computação I, desenvolva um programa para ser rodado no terminal (ou no shell) que possui uma interface básica com o usuário. O programa deve ter um menu e só pode ser encerrado quando o usuário digitar a opção de sair.

Temas sugeridos:

1. Crie um programa para implementar um jogo de Batalha Naval.
    a. O programa deve ter um menu que permite jogar (com pelo menos 2 opções – jogar contra o computador, jogar contra outro usuário), ver pontuação (de acordo com o nível), carregar um jogo que já havia sido iniciado e sair.
    b. A opção de jogar deve interagir com o usuário como um jogo de batalha naval, inicialmente pedindo as posições dos navios e mostrando o tabuleiro na forma de matriz e indicando as linhas e colunas. Na opção de jogar contra outro usuário, ambos os usuários vão jogar na mesma tela (mesmo terminal), não há comunicação entre computadores, ou entre interfaces diferentes. Nesse caso, você deve pedir as posições dos navios de ambos usuários. Os navios podem ter diferentes tamanhos e podem estar na horizontal ou na vertical. O seu programa deve perguntar ao usuário qual a posição (linha e coluna) do tabuleiro que deseja abrir. A medida que o jogo for avançando, o usuário deve ser capaz de ver no tabuleiro as posições disponíveis, as tentativas que resultaram em água e as tentativas que resultaram em acertos. O vencedor é aquele que acerta todos os navios completos.
    c. O tabuleiro deve ser representado como uma lista de listas (ou tupla de tuplas), ou seja, uma matriz com pelo menos 10 linhas e 12 colunas.
    d. Navios sugeridos: 1 porta-aviões (1x5 ou 5x1); 2 cruzadores (1x4 ou 4x1); 2 hidro-avião (1x3 ou 3x1) ; 3 rebocadores (1x2 ou 2x1); 3 submarinos (1x1).
    e. Utilize o módulo random para sortear as posições dos navios para o computador e para sortear as jogadas do computador. Não é necessário acrescentar nenhuma inteligência ao jogo, mas o computador não pode tentar acertar uma posição que já foi aberta. Lembre-se que é interessante tentar posições na horizontal e na vertical vizinhas a de um acerto, caso o navio acertado ainda não esteja completo, mas não é obrigatório.
    f. Utilize arquivos para permitir que o estado do jogo seja salvo e para salvar as pontuações. Isto é, o seu programa deve ser capaz de carregar um jogo já começado.

2. Crie um programa para implementar a agenda de compromissos de dois meses de um usuário.
    a. O programa deve ter um menu que permite ver os compromissos, adicionar compromissos, alterar um ou mais compromissos, apagar um ou mais compromissos (com a opção de apagar todos os compromissos do mês, todos os compromissos de uma semana ou todos os compromissos de um dia), carregar agenda de compromissos salva (pelo menos dois meses devem ser salvos).
    b. O usuário deve poder escolher os meses que quer salvos e deve existir a possibilidade de apagar um mês para adicionar outro. Garanta que o usuário não escolha o mesmo mês duas vezes.
    c. Se o programa perceber que não há nenhum mês salvo, ele deve pedir os meses desejados (e ano) ao usuário. Para descobrir os dias da semana para um mês novo, utilize o módulo calendar do Python: depois do import calendar, utilize a função calendar.setfirstweekday(6), para definir que o primeiro dia da semana é domingo, conforme pedido, e, em seguida, use calendar.monthcalendar(ano, mes). A função calendar.monthcalendar(ano, mes) retorna uma lista de listas com os dias do mês já separados em semanas. Outra função que pode ajudar é a calendar.weekday(ano, mês, dia) que retorna 0 se o dia da semana for segunda-feira, 1 se for terça e assim por diante até o número 6, que representa domingo.
    d. Para cada uma das opções, é importante mostrar o calendário na tela no formato de uma matriz (como usual, separando por semanas, mostrando o dia da semana, onde domingo deve ser o primeiro dia). Ao mostrar o calendário, os dias com compromissos devem estar destacados (o dia com compromisso pode ter um asterisco ao lado do número, por exemplo).
    e. Se o usuário desejar ver os compromissos, seu programa deve informar os meses salvos, mostrar o calendário, e perguntar qual o dia que o usuário deseja ver. Ao mostrar os compromissos do dia, apresente em ordem dependendo da hora: compromissos mais cedo primeiro. As opções de alterar um ou mais compromissos e apagar um ou mais compromissos também devem seguir essa lógica: informar os meses salvos, e mostrar o calendário antes de efetivamente alterar ou apagar um compromisso.
    f. A lista de compromissos deve ser representada como uma lista de listas (ou tupla de tuplas), onde cada sublista representa um dia do mês e pode ter diversas strings que indicam os compromissos e os horários dos compromissos.
    g. Após cada operação, sempre volte ao menu principal (ou dê uma opção ao usuário para que ele volte ao menu principal).
    h. Utilize arquivos para permitir que dois meses de compromissos sejam salvos.

O tema do trabalho também pode ser sugerido pelo aluno, e é encorajado que vocês pensem em um tema que gostem, mas o programa deve contemplar: funções, interação com o usuário, laços de repetição, condicionais, lista de listas (ou tupla de tuplas ou lista de dicionários ou dicionário de

listas), a utilização de pelo menos um módulo e utilização de arquivos (para salvar o estado do programa). **Caso você não goste dos temas sugeridos e tenha uma ideia do que quer fazer, fale com a professora antes de começar a fazer o trabalho para que ela avalie se o tema está de acordo com o que foi pedido.**

Algumas exigências:

- Comente o código;
- Utilize funções para modularizar o seu código. As funções devem ter argumentos e valores de retorno (não é obrigatório, nem necessário, que todas as funções tenham argumentos e valores de retorno, mas é importante mostrar que você sabe utilizar esse recurso e que você sabe em quais situações ele é útil).
- Não utilize variáveis globais.
- Utilize nomes de variáveis e funções significativos.
- Lembre-se de verificar os valores passados pelo usuário ao fazer a interação (não se preocupe em verificar todas as possibilidades possíveis, mas faça uma verificação mínima).
- Utilize a variável __name__ (vídeo 34) para incluir testes no seu código.
- Utilize pelo menos uma função de um módulo.
- O trabalho deve ser apresentado para a professora através de uma videoconferência ou através de um vídeo onde o aluno explica o trabalho. O aluno não precisa aparecer no vídeo ou na chamada, somente áudio é suficiente. A apresentação deve ter em torno de 15 minutos e o aluno deve falar sobre os pontos principais do trabalho – o mais importante é explicar a lógica de programação. Não é necessário entrar em detalhes sobre todas as funções, basta explicar a lógica das funções mais importantes.

O código (um ou mais arquivos .py) deve ser entregue através do PoliMoodle. Os alunos que escolherem gravar o vídeo podem fazer o upload do vídeo para o youtube (ou mesmo para o google drive ou similares) e informar o link no comentário do PoliMoodle.

A organização do menu e do programa em si fica a critério do aluno. Além da lista de listas (ou tupla de tuplas ou lista de dicionários ou dicionário de listas), outros tipos de dados também podem ser utilizados, mas não são obrigatórios. O mesmo é válido para módulos, não é obrigatório utilizar só um módulo. Métodos e funções pré-prontas do Python também estão liberados. Não defina funções dentro de outras funções.

A matéria necessária para implementar o trabalho é aquela vista até a semana 7. A matéria da semana 8 (tratamento de exceção), não é obrigatória para o trabalho, mas ajuda a tratar possíveis erros que possam surgir na interação com o usuário.

Atenção! Todos os conceitos vistos ao longo do curso são considerados no trabalho, inclusive as boas práticas de programação e a usabilidade do programa. O código desenvolvido no trabalho deve ser flexível e permitir a sua utilização como um módulo – isto é, você deve permitir que seu código possa ser importando e que as funções possam ser utilizadas a partir de outro código (para isso, é

necessário dividir o código em funções e usar funções que recebem argumentos e retornam valores, conforme pedido acima).