Nivan's Maze

membros:

1) Vitor Modesto Leitão
2) Italo Felipe de Andrade
3) Ian Gabriel Braga Trinta
4) Frederick Almeida Lopes

github: https://github.com/FrederickAlmeida/projeto-ip

A organização do código pode ser dividida em:

1) imports
   Usamos a biblioteca pygame para fazer a parte principal do jogo, como desenhar. Objetos na tela e fazê-los interagir entre si, com colisões e armazenando essas informações.
   Também fizemos uso da biblioteca sys, para fazer com que se o usuário apertasse no botão de sair, a tela fechasse.
2) Informações iniciais
   a segunda parte é uma pequena parte do código onde declaramos variáveis como fonte, e algumas outras para fazer a tela de game over
3) Classes
   fizemos 3 classes para o jogo, item é a classe que será herdada pelos fragmentos colecionáveis, que são o objetivo do jogo, ao ser chamada, precisa receber os valores do nome, tipo posição x e posição y, o valor de tipo vai ser usada para diferenciar cada fragmento, pois achamos que seria uma boa pro visual do jogo se cada fragmento tivesse sprites diferentes, e as posições são para desenhar o objeto no plano cartesiano. item_modificador_vida são os objetos que vão mudar a vida do usuário, achamos que seria coerente fazer em uma classe separado pois não há muitas semelhanças com os fragmentos, assim como a classe item, essa também recebe os valores de tipo, posição x e y, porém o tipo nessa classe tem uma funcionalidade a mais, achamos que o jogo ficaria menos fácil se cada armadilha e vida tivessem valores diferentes, para que o jogador não pudesse saber previamente o que iria acontecer se ele colidisse com uma armadilha ou poção, ou poção especial, trazendo mais emoção para o jogo, e para isso, a variável tipo “caiu como uma luva”, pois com isso nos colocamos if’s no código para analisar qual tipo seria, e com isso, declaramos atributos diferentes. Além disso, também recebe o valor de nome, que serve apenas para identificar se o objeto é uma poção, armadilha, ou poção especial. A última classe usada foi personagens, que não precisa receber nenhum parâmetro, pois nós já sabíamos previamente em que posição o personagem ia nascer, e seu tipo. Dentro da classe há funções para a movimentação dele e outra que faz ele voltar ao início, pois caso o personagem colida com alguma parede, ele voltará para o início

4) Jogo em si
   O jogo em si ficou todo dentro de um laço while, dentro dele, nós desenhamos todos os objetos, como personagem, armadilha, poções e as paredes, que são

desenhadas como linhas vermelhas, caso o usuário colida com alguma parede, foi criada uma função colidiu, que vai chegar na lista de paredes se ele colidiu com alguma delas, caso ela retorne True, é usada a função dentro da classe personagens, voltar_inicio, que muda as coordenadas para a de partida, caso ele colida com algum objeto que altere vida, a variavel desenhar_objeto_x é dada como False, fazendo com que ela não seja mais desenhada, pois caso um item seja consumido, não haverá chance de desenhá-lo novamente e, após isso, é aplicado o efeito a vida do personagem, que depende do tipo de item que ele colidiu, caso ele colida com um fragmento, o efeito será em suma o mesmo, mas com a diferença de que a variável que armazena a quantidade de fragmentos coletados será somada com 1, caso chegue a 3(quantidade total) o jogo acaba.

Ferramentas

nosso trabalho foi feito utilizando as bibliotecas pygame e sys, onde cada uma foi utilizada foi dito anteriormente, todos os membros do grupo utilizaram o VScode para programar, e usamos o GitHub para armazenar e compartilhar o código, já no tocante a comunicação e organização de reuniões, usamos o discord.

Divisão de trabalhos

Frederick: Responsável por programas as classes usadas no jogo e elaboração da história
Italo: Responsável pela partida gráfica e união dos códigos dos membros do grupo
Ian: Barra de vida e tela de game over
Vitor: Colisões, efeitos a serem aplicados e união dos códigos dos membros do grupo

É importante deixar claro que todos participaram igualmente da criação do jogo, desde a parte de idealização, até os últimos detalhes, não tivemos nenhum problema de organização na questão de divisão de tarefas.

Conceitos das aulas

Os principais conceitos que usamos foram a lógica para programação, que nos ajudou a aprender Pygame e POO em pouco tempo e conseguir utilizá-la da melhor forma que conseguimos. Laços de repetição, uma vez que grande parte do jogo é feito dentro de um while True. Condicionais, pois precisamos usar em todo o projeto, começando pelas classes, para analisar qual é o tipo do objeto, também para “fechar” o jogo, para analisar se o personagem colidiu e para conferir se é pra desenhar X objeto. Funções e listas também foram dois assuntos essenciais para o funcionamento do código da forma que idealizamos, por conta da função colidiu, que é usada muitas vezes durante o jogo, tal função que depende totalmente da lista de paredes(linhas).

Desafios

Houve diversos desafios na construção do jogo, como conciliar o tempo entre programação do jogo e estudo das outras cadeiras do período, também tivemos problemas de bugs

pontuais, mas conseguimos resolver trabalhando em equipe. Além disso, o problema mais difícil de solucionar foi juntar os códigos que foram feitos separadamente, pois é sempre mais difícil ler um código feito por outra pessoa, além disso, surgiram bugs durante o processo, mas conseguimos resolvê-los. Aprendemos muitas coisas durante a jornada de criação do jogo, mas com elas vieram aprendizados essenciais para o mercado de trabalho, como gerenciamento de pessoas e tempo, uma vez que o trabalho em equipe se faz necessário na área de computação. Apesar dos desafios apontados, não tivemos nenhum erro muito grande durante o projeto, apenas bugs pontuais que foram resolvidos rapidamente.