

CADERNO DE RESPOSTAS DA ATIVIDADE PRÁTICA DE:

Linguagem de Programação APlicada

ALUNO: André vieira dos santos 2792750

**ANO**

**2024**

Caderno de Resposta Elaborado por:

Prof. MSc. Renan Portela Jorge

|  |
| --- |
| **Questão 01 – Aula Prática 01** |
| **Dado o seguinte trecho de código da aula prática 01, indique as chamadas de método em que é feito o desenho na tela e qual a necessidade do método pygame.display.flip() para visualização das superfícies.** |
| RESPOSTA: Os métodos utilizados para chamar os desenhos são  window.blit(source=bg\_surf, dest=(bg\_rect)) esse método realiza o desenho do plano de fundo na superfície  window.blit(source=player\_surf, dest=(player\_rect)) esse método realiza o desenho do objeto player na superfície após realizar o desenho do background  O método pygame.display.flip necessita para que cada execução do for seja atualizado a posição do objeto seja desenhado novamente o bg\_surf e player\_surf na na nova posição na tela do programa. |

|  |
| --- |
| **Questão 02 – Aula Prática 02** |
| **Na aula prática 02, começamos a construir o diagrama UML do nosso jogo (figura abaixo). A partir desse digrama explique a relação entre Level, Player e EntityFactory e de que maneira objetos da classe player não compõe EntityFactory, mas sim Level.**  Diagrama, Desenho técnico  Descrição gerada automaticamente |
| RESPOSTA: (INSERIR RESPOSTA AQUI)  Foi criado a Classe EntityFactory para que não sobrecarregue a classe level, a Classe EntityFactory é responsável pela criação de cada Player, Enemy e Background de acordo com o level selecionado, retirando essa responsabilidade da classe Level de estar criando os objetos. |

|  |
| --- |
| **Questão 03 – Aula Prática 03** |
| **Na aula prática 03, instanciou-se 14 objetos da classe background que foram inseridos dentro de uma lista de entidades que compõe o objeto Level1. Explica com suas próprias palavras como se alcançou o efeito de profundidade no cenário. Na sua explicação deve apresentar o motivo das 14 instancias do objeto background.** |
| RESPOSTA: Na classe BACKGROUND no método move foi instanciado o efeito de Parallax movimentação no cenário movendo para esquerda com o código abaixo  Foi criado uma variável chamara ENTITY\_SPEED na classe const, nessa variável foi definido que cada objeto se move em velocidade diferente ao executar o método \_\_init\_\_ na classe leve  Na classe ENTITY\_SPEED foi definido que cada objeto se move em uma velocidade gerando o efeito Parallax.  def move(self):  self.rect.centerx -= ENTITY\_SPEED[self.name]  if self.rect.right <= 0:  self.rect.left = WIN\_WIDTH |
|  |

|  |
| --- |
| **Questão 04 – Aula Prática 04** |
| **Na aula prática 04, utilizou-se da estratégia de instanciação de inimigos via evento temporal dentro da classe Level. De que maneira, os eventos são verificados? Apresente também é em que momento se atrela o intervalo temporal com o id do evento** |
| RESPOSTA: Os Eventos são verificados dentro do Loop principal do jogo que é no método run() da classe Level  O intervalo temporal atrelado ao ID do evento está no momento da instanciação da classe Level na linha pygame.time.set.timer(event\_ememy, 4000) onde está configurado para um tempo de 4 segundos para que um novo evento de criação de inimigos seja gerado conforme especificado no EVENT\_EMENY  Então no loop principal do jogo sempre que um inimigo foi detectaco , um novo inimigo será adicionado a lista de entidade do jogo, e isso no intervalo de tempo dado pelo temporizador configurado durante a iniciação da classe Level. |