

CADERNO DE RESPOSTAS DA ATIVIDADE PRÁTICA DE:

Linguagem de Programação APlicada

ALUNO: Carlos Henrique monnerat quintanilha

RU: 4328237

**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: |

|  |
| --- |
| **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:  A relação entre Level, Player e EntityFactory é de composição.  EntityFactory funciona como auxiliar para criar instâncias de objetos da Classe Player, Enemy e Background. Dessa forma, os objetos da classe Player criados por EntityFactory não pertencem a EntityFactory e sim a Level, assim Player não compõe EntityFactory. |

|  |
| --- |
| **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:  O efeito de profundidade no cenário foi alcançado pois cada objeto inserido representa uma camada, em que o primeiro representa o objeto mais distante e o último representa o mais próximo.  O jogo renderiza primeiro o objeto mais distante e só depois renderiza os mais próximos, por isso as 14 instâncias foram necessarias para criar o efeito de profundidade da melhor forma. |

|  |
| --- |
| **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 for event in pygame.event.get(). Este loop examina todos os eventos desde a última interação continuamente.  O intervalo temporal se atrela com o id do evento na linha de código:  pygame.time.set\_timer(EVENT\_ENEMY, 4000) |