Exame

NOME: Nr.MEC:

Questões de escolha múltipla (1 a 20): **responda na grelha**, assinalando uma opção por pergunta (pretende-se a opção verdadeira e, havendo várias que possam ser consideradas verdadeiras, pretende-se a mais abrangente); as não-respostas valem zero; **respostas erradas descontam** ¼ da cotação; as respostas assinaladas de forma ambígua serão consideradas não-respostas.
Questões 21 e 22: responder em folha separada, identificada.

Grelha de respostas (perguntas 1 a 20):

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
a)																				
b)																				
c)																				
d)																				
e)																				

P1.

Os métodos ágeis de desenvolvimento de software preconizam a manutenção de uma relação aberta, transparente e cooperante com o cliente através de práticas tais como:

a) Participação do cliente na definição da arquitetura da solução.
b) Participação do cliente na priorização dos casos de utilização.
c) Monitorização partilhada da velocidade de desenvolvimento (pontos realizados por iteração).
d) Alteração frequente dos requisitos do produto.
e) Envolver o cliente nos testes diários de aceitação.

P2.

O método *Unified Process* prevê quatro fases principais no desenvolvimento do projeto. Cada qual tem um grande objetivo a atingir (*milestone*) para se poder avançar, por esta ordem:

a) 1/ Preparação do documento de Visão; 2/ Demonstração de protótipos exploratórios; 3/ Produto implementado; 4/ Testes no cliente.
b) 1/ Definição dos casos de utilização; 2/ implementação do protótipo exploratório; 3/ definição da arquitetura; 4/ Implementação do produto concluída.
c) 1/ Plano para o projeto definido; 2/ Análise de requisitos terminada; 3/ Arquitetura técnica do produto definida; 4/ Implementação da solução concluída.
d) 1/ Decisão de avançar ou parar o projeto; 2/ Arquitetura técnica definida e validada; 3/ Funcionalidades da primeira versão do produto implementadas; 4/ Solução instalada e aceite pelo cliente.
e) 1/ Definição das histórias de utilização ; 2/ Gestão ativa do *backlog*; 3/ Arquitetura técnica validada; 4/ Implementação da solução concluída.

P3.

O método Unified Process é orientado aos casos de utilização (CaU), porque neste método:

a) A primeira tarefa do SDLC é o levantamento e especificação de CaU, incluindo cenários típicos e alternativos.
b) Os CaU estabelecem uma divisão funcional do sistema que é usada ao longo do desenvolvimento do projeto, na análise, no desenho e nos testes.
c) A UML e os Diagramas de Casos de Utilização são usados para descrever o projeto.
d) É recomendada a identificação de personas e a exploração de cenários de uso através de histórias ("user stories").
e) Todas as hipóteses anteriores estão corretas.

P4.

Considere o requisito a seguir formulado, relativo à operação de uma loja *online*, no contexto da análise de requisitos de um sistema de software:

R_i: "Os utilizadores devem autenticar-se na Loja (*login*) para realizar todas as operações, exceto para a consulta dos artigos em promoção."

a) É um requisito funcional, mas inadequado, porque só abrange os artigos em promoção.
b) É um exemplo de um requisito não funcional, relacionado com a segurança do sistema.
c) É um exemplo de um requisito não funcional, relacionado com a fiabilidade do sistema.
d) É um exemplo de um atributo de qualidade, relacionado com a usabilidade do sistema.
e) Não é um bom exemplo de um requisito, porque não é verificável.

P5.

Os conceitos de Ator e Persona são usados na análise de sistemas para descrever cenários de uso de um sistema.

a) Uma Persona é uma forma mais específica de um Ator (relação de herança).
b) A Persona é uma instância/concretização de um Ator.
c) A Persona e o Actor podem ser representados no Diagrama de Casos de Utilização.
d) Persona e Ator não estão associados, pois a Persona não pode ser usada para modelar um sistema externo, e o Ator sim.
e) A inclusão de Personas na Análise é desaconselhada porque o conceito não existe na UML.

P6.

Na SCRUM, há lugar à gestão ativa da pilha de trabalho (*backlog*). Neste contexto, que propriedades são próprias do *backlog*?

a) Todos os itens de trabalho são classificados com uma nota, quanto à sua prioridade.
b) Cada item de trabalho corresponde a um caso de utilização.
c) As histórias de utilização (*user stories*) apresentam um nível de detalhe adequado para ser usadas como itens do *backlog*.
d) A pilha está ordenada, das funcionalidades com maior pontuação (no topo) para as que têm menos pontuação.
e) A posição de cada item de trabalho na pilha não deve ser alterada; apenas se "tiram" itens do topo para serem implementados.

P7.

A arquitetura trata da tomada das grandes decisões técnicas em relação ao sistema a desenvolver, tendo em conta os atributos de qualidade pretendidos. Um exemplo de um assunto/decisão de arquitetura é:

a) Especificar os cenários de interoperação com sistemas externos e as tecnologias selecionadas para os implementar.
b) Definir estratégias de distribuição de carga para garantir a disponibilidade do sistema em utilização contínua, com 1000 sessões simultâneas.
c) Definir os mecanismos técnicos para adaptar um sistema existente aos novos requisitos previstos no RGPD.
d) Todas as opções anteriores são corretas.
e) Nenhuma das opções anterior é correta.

P8.

Uma das principais razões para se utilizar métodos ágeis de desenvolvimento, em detrimentos dos métodos sequenciais, é a diminuição do risco do projeto. Que prática é decisiva para a mitigação do risco:

a) A ordem dos itens na pilha do *backlog* é imutável, tornando o projeto mais previsível.
b) Os projetos são mais pequenos e não se gasta tanto tempo em tarefas de coordenação e documentação.
c) O teste do software é deslocado da equipa de desenvolvimento para o Cliente.
d) A entrega frequente de valor diminui os problemas decorrentes de eventuais divergências na perceção dos requisitos.
e) O projeto está dividido em iterações autónomas e independentes.

P9.

O modelo do domínio é construído na Análise, para mapear os conceitos do universo de discurso.

a) As classes do modelo do domínio são as mesmas classes do código e por isso existe uma continuidade.
b) As classes do modelo do domínio captam a estrutura da informação; terão utilidade posterior para desenhar a base de dados, mas não o código.
c) Os conceitos identificados no modelo do domínio serão candidatos naturais a constituir classes, na fase de desenho.
d) A estrutura dos dados de um problema é muito volátil e, por isso, o modelo do domínio é provisório e vai ser alterado.
e) O modelo do domínio só é construído para projetos que requerem a utilização de bases de dados.

P10.

É possível avaliar a coerência entre diferentes diagramas de um modelo. Considerando a transposição do diagrama de objetos para o diagrama de classes, devemos garantir que:

a) Todos os atributos de um Objeto são representados como atributos de uma Classe.
b) Os atributos de um Objeto podem ser organizados e arrumados em diferentes Classes.
c) Os Objetos que não especificam o seu classificador/tipo, são modelados com Classes abstratas.
d) Tem de haver tantas Classes quantos os Objetos representados.
e) Não é possível derivar um diagrama de Classes de um diagrama de Objetos, porque são níveis de abstração distintos.

P11.

Considerando a capacidade expressiva do Diagrama 1:

a) Uma Empresa tem mais de um Departamento.
b) Um Escritório pode ser partilhado por várias Empresas.
c) Uma Empresa ou tem Escritórios ou tem Departamentos.
d) O Escritório tem um Empregado que é responsável por aquele.
e) Um Departamento tem sempre Empregados.

P12.

Que alterações seria necessário fazer ao Diagrama 1 para captar o requisito: "Um Empregado participou num projeto por um período de tempo designado."

a) A classe Projeto deve ter dois atributos adicionais, para indicar o início e o fim do período.
b) A classe Empregado deve ter dois atributos adicionais, para indicar o início e o fim do período.
c) As classes Projeto e Empregado devem especializar uma classe abstrata com a data de início e fim do período.
d) A associação de participação, entre Projeto e Empregado, deve ser caraterizada com uma classe de associação.
e) Essa informação não é suscetível de ser captada num modelo UML.

P13.

Relativamente ao Diagrama 1:

a) Não é possível realizar projetos, porque a classe Projeto não pode ser instanciada.
b) Um Departamento não pode ser responsável por projetos internos e por projetos externos.
c) Os projetos externos têm um gestor designado.
d) Os Empregados têm Escritórios dos quais são responsáveis.
e) Cada projeto interno tem um Departamento responsável.

P14.

O Diagrama 2 utiliza o conceito de estereótipo da UML (*stereotype*) na relação "extend".

a) É sempre possível substituir o estereótipo "extend" por "include", alterando a ordem na apresentação do diagrama.
b) O estereótipo "extend" tem associados pontos de extensão do caso de utilização, que podem ou não se visualizados.
c) O estereótipo "extend" é acessório, não altera a natureza da relação de dependência.
d) O fluxo de utilização incluído em "Criar diagnóstico" inclui sempre a sequência em "Consultar exames de imagiologia"
e) Todas as hipóteses estão corretas.

P15.

Relativamente ao modelo representado no Diagrama 2

a) Está incompleto: não inclui a fronteira do sistema.
b) Utiliza a relação de hierarquia entre atores.
c) O "Sistema de Imagiologia Médica" não é um ator, porque é um sistema e não um utilizador.
d) Deveria relacionar o caso de utilização "Pesquisar utente" com "Marcar consulta".
e) Todas as afirmações estão corretas.

P16.

Nos elementos modelados no Diagrama 2 há um que está claramente desajustado. Trata-se de:

a) O ator Supervisor: este papel já está previsto no Secretariado.
b) O ator Secretariado: está englobado pelo papel do Supervisor.
c) O caso de utilização "consultar exames imagiologia": é uma parte dentro do caso de utilização "criar diagnóstico".

d) A associação entre os atores Supervisor e Secretariado: deveria ser uma dependência marcada com <<extend>>.
e) O caso de utilização "Dados do utente": não tem fluxo subjacente.

P17.

Relativamente ao processo de trabalho descrito no Diagrama 3:
a) Um trabalho é avaliado logo que é entregue.
b) Um trabalho não pode ser entregue depois do fim do prazo.
c) Os alunos que entregaram e os que desistiram recebem feedback do docente.
d) O trabalho tem diferentes estados, ao longo da atividade.
e) O trabalho tem diferentes estados, mas estão mal atribuídos às partições.

P18.

Em que fase do SDLC é mais natural que se construa um resultado como o Diagrama 3:
a) Na Análise, para caraterizar processos de trabalho existentes ou os novos processos pretendidos.
b) Na Análise, para fazer o levantamento dos atores.
c) Na Análise, para fazer o levantamento dos casos de utilização e cenários subjacentes.
d) Na Implementação, para clarificar os algoritmos que os métodos das classes devem codificar.
e) Na Implementação, para mapear os objetos de informação trocados entre classes.

P19.

Relativamente ao tipo de diagrama ilustrado no Diagrama 3 e à sua relação com outros diagramas da UML:
a) Os estados anotados nos objetos podem ser visualizados num diagrama de estados.
b) As ações modeladas são também casos de utilização (Lançar trabalho, Realizar Trabalho,…).
c) As partições de um Diagrama de Atividades são os atores do sistema.
d) Os eventos temporais correspondem à mensagem inicial num diagrama de sequência.
e) Todas as hipóteses anteriores são corretas.

P20. [questão de desenvolvimento]

O Manifesto para o Desenvolvimento Ágil de Software apresenta 12 princípios, nos quais se inclui esta ideia: "os processos ágeis fomentam um **desenvolvimento sustentado**. Os promotores, a equipa técnica e os utilizadores devem ser capazes de manter um ritmo [de trabalho/progresso] contínuo".
Explique o que são os "processos ágeis" e como é que contribuem para se atingir o preceito do "desenvolvimento sustentado" num projeto de sistemas de informação.

P21. [questão de desenvolvimento]

Considere o trecho de código seguinte, em Java, com omissões.
a) Apresente um diagrama de classes para visualizar, de forma detalhada, a informação estrutural da classe ConversorMoedas e das suas dependências diretas.
b) Apresente um diagrama de sequência para representar a interação entre objetos que ocorre quando é invocado o método ConversorMoedas#**converter()**.

```
3   public class ConversorMoedas { /// algum codigo omitido ///
4
5       private TabelaDeCambio tabelaCambio;
6
7       /**
8        * converte uma quantia monetária para o correspondente noutra moeda
9        * @param origem valor base a converter
10       * @param codigoMoedaDest codigo internacional da moeda pretendida (EUR, USD, GBP..)
11       * @return o valor convertido para a nova moeda
12       */
13      public Quantia converter(Quantia origem, String codigoMoedaDest) {
14          Quantia convertido;
15
16          if (tabelaCambio.existe(origem.getCodigoInternacional()) &&
17                  tabelaCambio.existe(codigoMoedaDest)) {
18              String codigoMoedaOrigem = origem.getCodigoInternacional();
19              double cambio = tabelaCambio.procurarTaxa(codigoMoedaOrigem, codigoMoedaDest);
20              double valorInicial = origem.getValor();
21              convertido = new Quantia(codigoMoedaDest,  valorInicial * cambio);
22          } else {
23              throw new IllegalArgumentException( "Códigos de moeda não reconhecidos.");
24          }
25          return convertido;
26      }
27  }
```

FOLHA DE DIAGRAMAS

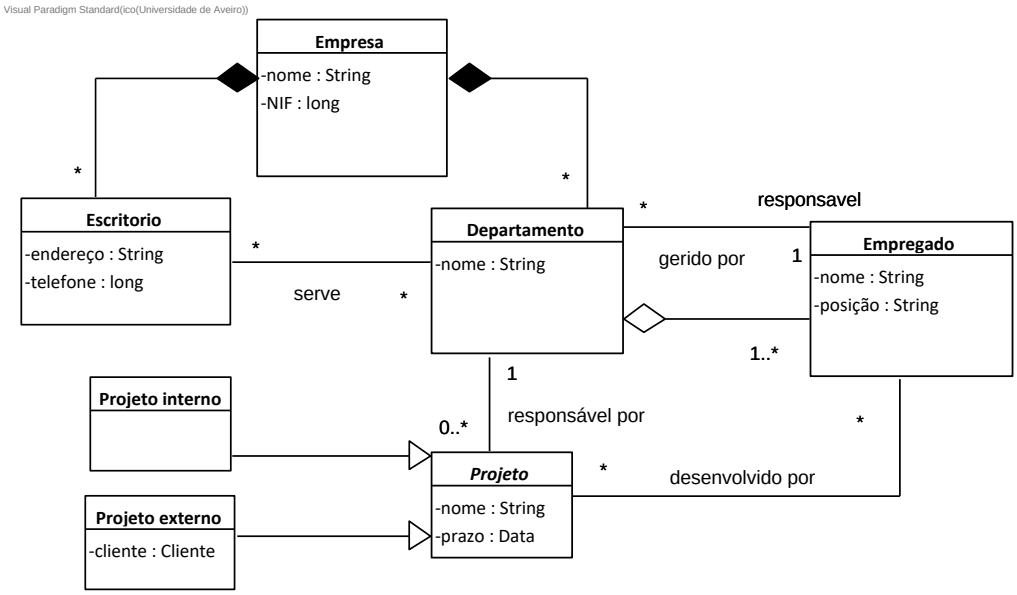

Diagrama 1- Conceitos associados à organização interna de uma empresa.

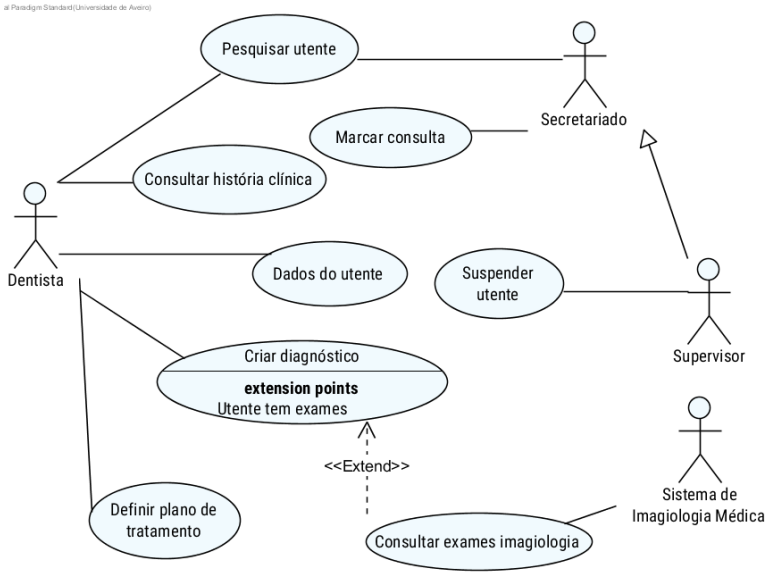

Diagrama 2- Cenários de utilização associados a uma clínica dentária.

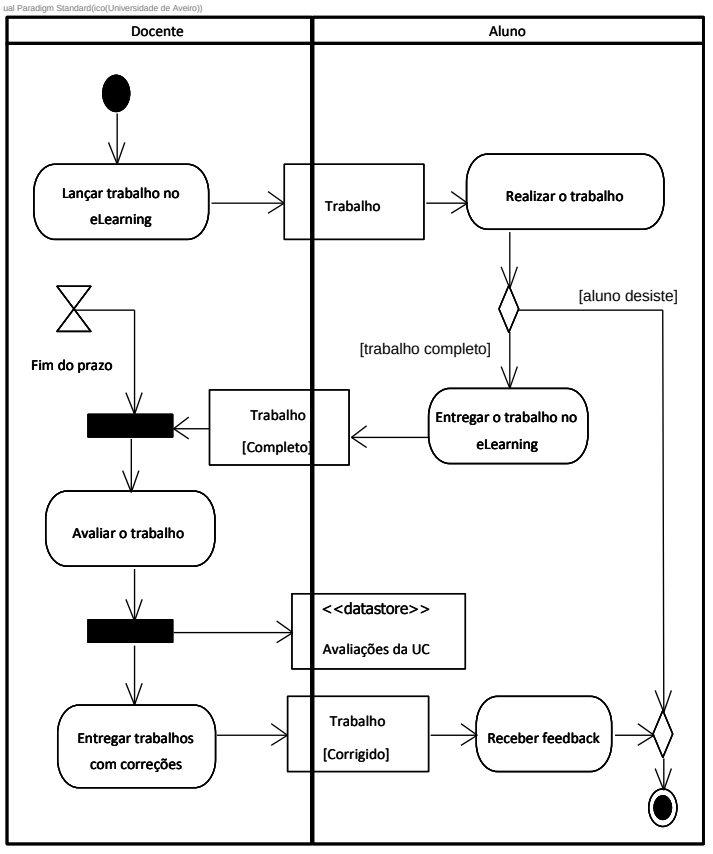

Diagrama 3: Atividades associadas à realização de trabalhos académicos.