Fundação CECIERJ - Vice Presidência de Educação Superior a Distância

Curso de Tecnologia em Sistemas de Computação
Disciplina: Engenharia de Software
AP3 2° semestre de 2014.

Nome –

Assinatura –

Observações:

1. Prova sem consulta e sem uso de máquina de calcular.
2. Use caneta para preencher o seu nome e assinar nas folhas de questões e nas folhas de respostas.
3. Você pode usar lápis para responder as questões.
4. Ao final da prova devolva as folhas de questões e as de respostas.
5. Todas as respostas devem ser transcritas nas folhas de respostas. As respostas nas folhas de questões não serão corrigidas.

---

1) Explique o conceito de "projeto" no contexto do gerenciamento de projetos de software. Dica: existem três características que devem estar presentes para que um trabalho seja considerado um projeto - defina o conceito de projeto a partir destas características. (Valor: 2,0 pontos).

Um projeto é um trabalho realizado dentro de uma organização que tem por objetivo gerar um produto ou serviço único, é um esforço temporário (no sentido de ter um prazo para acabar) e é realizado de forma incremental.

2) Porque dizemos que na implementação de um projeto software é mais importante que um programa seja simples do que elegante? Relacione este tema com a necessidade de um programador desenvolver um sistema de forma a simplificar o trabalho de outros programadores que poderão alterar este sistema no futuro. (Valor: 2,0 pontos).

Grande parte do esforço no desenvolvimento de um produto de software é investido em sua manutenção. Sendo assim, o produto deve ser construído de forma a tornar sua manutenção mais simples e barata. Por isto dizemos que um programador deve programar para outro programador, que venha a sucede-lo na manutenção do projeto.

Códigos simples são mais fáceis de entender, simplificando assim o trabalho de outros programadores.

3) Diferentes documentos devem ser criados ao longo do processo de desenvolvimento que, por sua vez, pode assumir diferentes configurações dependendo do nível de maturidade da organização. Desta forma, indique 2 documentos que devem ser criados por uma organização de software nível C no MPS-BR, explicando seus objetivos e sugerindo quais são seus usuários em potencial. (valor: 2,0 pontos; máximo: 10 linhas)

Qualquer documento produzido nas áreas de processo previstas no nível C (veja o material do curso). Por exemplo,

gerencia de requisitos: especificação de requisitos descrevendo os requisitos do software usado pelos desenvolvedores, testadores e clientes.

gerencia de projeto: plano de projeto descrevendo os diferentes modelos e artefatos que deverão ser criados ao longo do projeto usado pelo gerente, desenvolvedores e testadores

medição: conjunto de medidas que devem ser coletadas e consideradas para observar o "comportamento" do software e do processo usado pelos desenvolvedores e gerentes

gerencia de configuração: plano de configuração descrevendo a estrutura organizacional dos diferentes artefatos, módulos e componentes. Usado pelos desenvolvedores, testadores e gerentes.

gerencia de aquisição: plano de aquisição relacionado ao processo de desenvolvimento. Utilizado pelos gerentes do projeto.

garantia da qualidade: plano de qualidade do software estabelecendo os critérios e comportamentos esperados para o software usado pelos desenvolvedores, testadores, gerentes e stakeholders

gestão de portfólio de projetos: documento descrevendo os diferentes projetos em andamento e suas características. Utilizado pelos gerentes de projeto.

dentre outros.

4) "A manutenção corretiva de software não pode ser evitada, independente de como o software é construído". Esta afirmação é verdadeira ou falsa? Justifique sua resposta indicando os tipos de manutenção existentes e apresentando 3 fatores que podem afetar o esforço de manutenção de software juntamente com 3 responsabilidades da equipe de manutenção. (valor: 2,0 pontos).

A afirmação é falsa. Atividades de manutenção corretiva referem-se a correção de defeitos em software. Portanto, o uso de abordagens de desenvolvimento que privilegiem a prevenção e remoção de defeitos pode levar a eliminação de manutenção corretiva após a entrega do software.

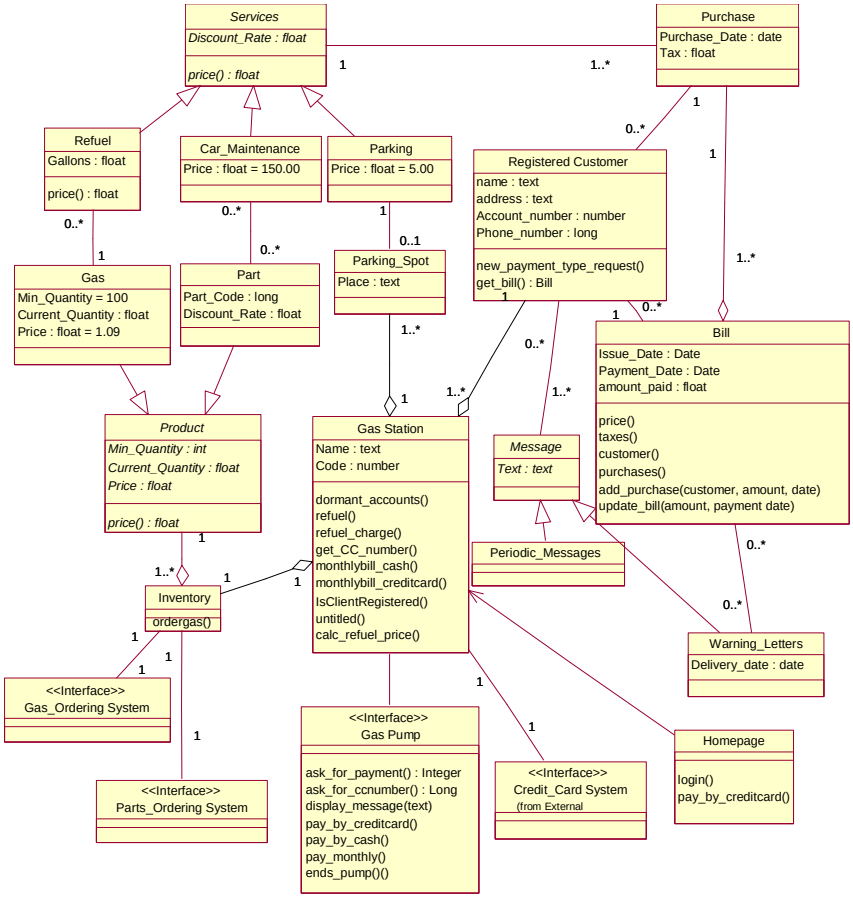
As atividades executadas podem ser de:
Correções:Corrige um defeito – i.e. uma discrepância entre o comportamento requerido para um produto/aplicação e o comportamento observado
Melhorias:Implementam uma mudança para o sistema que modifica seu comportamento ou implementação.Melhorias podem ser: Troca de requisitos (Manutenção Perfectiva), Adiciona um novo requisito ao sistema (Manutenção Adaptativa) e Troca a implementação mas não o requisito (Manutenção Preventiva)

Fatores: Tipo de aplicação, Novidade do sistema, Rotatividade e disponibilidade do pessoal de manutenção, Duração da vida útil do sistema, Dependência de um ambiente que se modifica, Característica de hardware, Qualidade do projeto, Qualidade do código, Qualidade da documentação, Qualidade dos testes

Responsabilidades da Equipe: entender o sistema, localizar informação na documentação do sistema,manter a documentação do sistema atualizada, estender as funções existentes para acomodar novos requisitos ou modificações nos requisitos, acrescentar novas funções para o sistema, encontrar a fonte de falhas ou problemas no sistema, localizar e corrigir faltas, responder questões sobre a forma como o sistema funciona, reestruturar o projeto e codificação dos componentes, reescrever o projeto e código dos componentes, apagar os projetos e códigos de componentes que não são mais úteis, gerenciar trocas para o sistema

5) Considerando o Diagrama de Classes abaixo, encontre os valores de NOC (numero de filhos), DIT (profundidade de herança) e CBO (Acoplamento entre objetos) para as classes identificadas na tabela e indique qual delas pode oferecer mais risco à qualidade do produto, explicando ao menos um problema que poderia ocorrer. Leve em consideração as estruturas hierárquicas para extração das medidas (valor: 2,0 pontos):

CLASSE	NOC	DIT	CBO	Propensa à Falha
Gas	0	1	2	
Bill	0	0	3	
Registered Customer	0	0	4	
Services	3	0	1	
Gas Station	0	0	6	Mais propensa a falha pelo critério CBO (é a mais dependente de outras classes). Uma falha em qualquer classe "ligada" a ela pode leva-la a falha.


Services
Discount_Rate : float
price() : float
Purchase
Purchase_Date : date
Tax : float
Refuel
Gallons : float
price() : float
Car_Maintenance
Price : float = 150.00
Parking
Price : float = 5.00
Registered Customer
name : text
address : text
Account_number : number
Phone_number : long
new_payment_type_request()
get_bill() : Bill
Gas
Min_Quantity = 100
Current_Quantity : float
Price : float = 1.09
Part
Part_Code : long
Discount_Rate : float
Parking_Spot
Place : text
Bill
Issue_Date : Date
Payment_Date : Date
amount_paid : float
price()
taxes()
customer()
purchases()
add_purchase(customer, amount, date)
update_bill(amount, payment date)
Product
Min_Quantity : int
Current_Quantity : float
Price : float
price() : float
Gas Station
Name : text
Code : number
dormant_accounts()
refuel()
refuel_charge()
get_CC_number()
monthlybill_cash()
monthlybill_creditcard()
IsClientRegistered()
untitled()
calc_refuel_price()
Message
Text : text
Periodic_Messages
Inventory
ordergas()
Warning_Letters
Delivery_date : date
<<Interface>>
Gas_Ordering System
<<Interface>>
Parts_Ordering System
<<Interface>>
Gas Pump
ask_for_payment() : Integer
ask_for_ccnumber() : Long
display_message(text)
pay_by_creditcard()
pay_by_cash()
pay_monthly()
ends_pump()()
<<Interface>>
Credit_Card System
(from External
Homepage
login()
pay_by_creditcard()