Anotações Curso DEV Carrefour – MÓDULO I

Pensamento Computacional

(aula1)Classificação do raciocinio logico

(aula2)

***Exemplo de algoritmo***

Media de alunos:

* Passo 1 - recebe os valores
* Passo 2 - media aritimetica
* Passo 3 - imprime o resultado
* Passo 4 - regra de aprovação

Lógica de Programação

(aula1)

Metacognição – Pensar como você pensa

Algoritmo – Sequência de passos

Função condicional – “Se existir algo, faça alguma coisa”

Fluxograma é uma ferramenta utilizada para representar graficamente o algoritmo, isto é, a sequência lógica e coerente do fluxo de dados.

Constante – Valor constante (não muda)

Variável – Valor muda

**Expressões aritméticas**

Utilizam operadores aritméticos e funções aritméticas envolvendo constantes e variáveis

|  |  |
| --- | --- |
| Operadores | Símbolos |
| Soma | + |
| Subtração | - |
| Multiplicação | \* |
| Divisão | / |
| Potenciação | ^ |
| Porcentagem | % |

**Operadores Relacionais**

São expressões compostas por outras expressões ou variáveis numéricas com operadores relacionais

As expressões relacionais retornam valores lógicos (verdadeiro/falso)

|  |  |
| --- | --- |
| Símbolo | Nome do Operador |
| > | Maior que |
| >= | Maior ou igual |
| < | Menor que |
| <= | Menor ou igual |
| == | Igualdade |
| != | Diferente De |

Concatenação

* É um termo usado em computação para designar a operação de unir o conteúdo de duas String\*

\*String é uma sequência de caracteres (entre aspas duplas)

Para concatenar mais informações é usado o &

Ex: “seu nome é” & nome

* Estruturas de Repetição

Consiste em repetir um comando até que seja definido um limite

(Exemplo está na pasta de exercícios das aulas)

* Linguagem de programação

A principal função é servir de um meio de comunicação entre computadores e humanos

*Classificação*

|  |
| --- |
| Alto Nível |
| Cuja sintaxe se aproxima mais da nossa linguagem e se distanciam mais da linguagem da maquina |
| Baixo Nível |
| É aquela que se aproxima mais da linguagem da maquina |

***Compiladas*** (c#, c++, etc)

O código fonte é executado diretamente pelo sistema operacional ou pelo processador, após ser traduzido por meio de um processo de compilação.

***Interpretadas*** (phyton, java script)

É uma linguagem de programação em que o código fonte é executado por um programa de computador chamado interpretador, que em seguida é executado pelo sistema operacional ou processador.

***Portugol*** – é uma pseudolinguagem que permite ao programador pensar no problema em si e não no equipamento que irá executar o algoritmo.

***Desvio condicional “se, senao”***

É utilizada a palavra “se”, a condição a ser testada entre parênteses e as instruções que devem ser executadas entre chaves caso o desvio seja verdadeiro.

**Ex:** se (media>7) {

Escreva (“Parabéns você foi aprovado”)}

Para que quebrar linha: \n

Desvio condicional “caso”

**Ex:** caso

escreva("1 - Abrir Netflix 2 - Abrir Amazon prime 3 - Abrir Hbo MAX 4 - Sair")

inteiro menu = 0

escreva("\n" + "Sua opção")

leia(menu)

escolha(menu)

{

caso 1:

escreva("ok abrir netflix")

pare ***(assim fazer caso 2,3,4 e 5)***

caso contrario:

escreva("Você deve escolher as opções 1,2,3,4 ou 5")

***Matrizes e Vetores***

Matriz é uma coleção de variáveis de mesmo topo, acessíveis com um único nome e armazenados continuamente na memória.

A individualização de cada variável de um vetor é feita através de índices.

Os vetores são matrizes de uma só dimensão.

Git e GitHub

* Git

Linus Torvalds – Criador do Git (e do Linux), de maneira colaborativa.

***Benefícios de aprender as duas tecnologias (Git e GitHub)***

* Controle de versão
* Armazenamentos em nuvem
* Trabalho em equipe
* Melhorar seu código
* Reconhecimento

*GUI* - Interface gráfica no Git

*CLI* – Comand Line Interface – Linhas de código

|  |  |
| --- | --- |
| Lista de Comandos Básicos Utilizados Dentro do Terminal do CMD | |
| dir | Comando para listar as pastas |
| cd | Possibilita que se navegue entre os diretórios/pastas |
| cd / | Volta para o início do terminal |
| cd .. | Caminho inverso (sai de uma pasta/diretório) |
| cls | Limpa o terminal |
| mkdir | Cria uma nova pasta/diretório |
| echo | Printa o que se escrever depois dele no terminal |
| Echo hello > hello.txt | Cria o arquivo com o nome hello |
| del | Deleta o que tem dentro do repositório/pasta |
| rmdir  rmdir workspace /S /Q | Remove o repositório com tudo dentro dele |
| ls | Lista as pastas dentro do repositório que você esta |

***Como o Git funciona por baixo dos planos***

SHA1 – Significa Secure Hash Algorith, é um conjunto de funções de hash criptográficas projetadas pela NSA.

Blobs – Armazenam metadados (bloco básico da composição) / Armazena o Sha1 do arquivo.

Tree – Armazenam Blobs, e monta toda a estrutura do arquivo.

Commit – Junta tudo e da sentido. Aponta pra uma árvore, para um parente (para o ultimo commit realizado antes dele) para um autor e uma mensagem.

***Chaves SSH e Tokens***

*Chave SSH*

Forma de estabelecer conexão segura e encriptada entre duas maquinas.

*Token de acesso pessoal*

Feito no site do GitHub e só pode ser visualidade uma vez no site, é necessário salvá-lo na máquina (está em documentos).

***Primeiros comandos do Git***

Git init – iniciar repositório

Git add – adiciona

Git commit – criar primeiro commit

Ls – lista todas as pastas

Mv – move a pasta

**Arquivo Untracked** – Git não tem conhecimento dele

**Tracked** – é dividido em Unmodified, Modified e Staged

*GitHub é o nosso repositório remoto*

*Resolvendo problema de versão de arquivo no GitHub* – Abre o arquivo e define qual versão é a que vai ser a definitiva (corrigida/mais atual)

SCRUM

**Gestão de Projetos**

***Tradicional X Ágil***

**Software Tradicional (waterfall):** só permite que o projeto avance quando uma fase está inteiramente completa

**Ágil:** software construído por partes (incremental) e cada parte executa-se em um clico (iterativo)

Tradicional

Escopo definido na fase inicial do projeto (preditivo)

Projeto é controlado por fases e marcos

Cliente só ver o software funcionando na fase final do projeto

Resistência a mudanças

Ágil

Escopo definido ao longo do projeto (adaptativo)

Projeto é controlado por funcionalidades entregues

Cliente pode ver o software funcionando na parte inicial do projeto

Mudanças constantes de acordo com feedbacks contínuos.

Em projetos tradicionais, você corre o risco de descobrir que estava errado depois de meses. Com o SCRUM, você descobre que estava errado em no máximo 30 dias.

* Scrum é um dos frameworks de gerenciamento de projetos ágeis
* Projetos usando equipes pequenas e multidisciplinares produzem os melhores resultados.

***Papeis do time SCRUM***

|  |  |  |
| --- | --- | --- |
| **Product Owner (PO)** | **Scrum Master (SM)** | **Time de desenvolvimento (DEV)** |

*PO – Product Owner*

* Define funcionalidades do software (backlog
* Garante que o time de desenvolvimento entenda os itens do backlog no nível necessário
* Prioriza as funcionalidades de acordo com o valor do negocio
* Representa a área de negócios

*SM – SCRUM Master*

* Age como facilitador
* Auxilia a equipe a remover impedimentos
* Auxilia o PO no planejamento e estimativas do backlog
* Garante uso correto do SCRUM
* Não é gerente de projetos
* Treina o time de autogerenciamento e interdisciplinaridade

*DEV – Time de Desenvolvedores*

* Possui habilidades suficientes para desenvolver, testar criar e desenha, ou seja, tudo que for necessário para entregar o software funcionando.

**Time Box –** Tempo máximo para fazer uma cerimônia ou Sprint

**Sprint -** corrida, arrancada

*É o principal evento do SCRUM*

*Duração de 1 sprint – 30 dias (ou menos)*

***Composição da sprint***

*Planejamento da sprint*

* Participa o PO e o time de desenvolvimentos

*Reuniões diárias*

* Geralmente é feia em pé para melhor efetividade
* Participa o PO os DEVS e o SCRUM Master
* Responde 3 perguntas: O que fez no dia anterior, o que está programado no dia e o que está em impedimento.

*Revisão da Sprint (Review)*

* Time DEV apresente para o PO o trabalho feito

*Retrospectiva da Sprint*

* Reunião da equipe par Lições Aprendidas

**Release Planning**

Liberação ou lançamento de software é o lançamento da nova versão oficial do produto de software. Cada vez que o produto é criado ou modificado, o fabricante e seus desenvolvedores decidem sobre como distribuir ou o novo produto ou a modificação às pessoas que o utilizam.

*O que é um release planning?*

De forma simplificada, o release planning é uma espécie de documento que integra todos os objetivos almejados em relação às medidas e ações que serão implementadas em uma organização, além de identificar os prazos em que elas devem ser concluídas.

*Existem 2 tipos de Releases Planning*

* Múltiplas Squads – vários times distintos que obedecem às mesmas regras técnicas
* De Projeto – PO possui demanda muito grande quebrando-a em várias estórias e em várias sprints

*Analisando o Escopo*

**Product Backlog**

* É composto por Épicos e Estórias

Épicos – Incremento sem muito detalhamento, ajuda a te direcionar dos caminhos que deve seguir

Estórias – Detalhamento dos épicos, um épico normalmente se divide em varias estórias, onde ficam descritos o que deve acontecer e suas regras de negócios.

Como se escreve uma Estória

* Nome da estória
* Descrição da estória (eu, como, quero, quando)
* Regras de negocio
* Tela
* KPI
* Tagueamento
* Critérios de Aceite

*Riscos Positivos e Negativos para projetos*

Positivos – muito ignorado nos projetos, porém um dos fatores de maiores ganhos no desenvolvimento de sistemas.

Negativos – itens que podem afetar o prazo, curso ou escopo de um projeto de maneira que pode acabar inviabilizando-o.

Papel do PO na transformação digital

Transformação Digital - É um processo no qual as empresas fazem uso da tecnologia para melhorar o desempenho, aumentar o alcance e garantir resultados melhores. É trabalhar junto com o cliente e ter uma mente focada no objetivo de atender o cliente da melhor forma possível.

***Estórias VS Tarefas***

Estória é um conjunto de tarefas e uma estória é um conjunto de tarefas

*Critério de Aceite*

É uma lista de critérios que precisam ser alcançados para que a User Story (Estória) atenda aos requisitos do usuário e seja aceita pelo Product Owner. Tem o objetivo de: Definir limites para as User Stories. Ajudas o PO a detalhar em alto nível o que é necessário para entregar valor ao cliente.

Planning Poker é a atividade de mensuração de esforço e complexidade das tarefas e estórias.

Stakeholder é uma pessoa ou grupo que legitima as ações de uma organização e que tem um papel direto ou indireto na gestão e resultados dessa mesma organização.

Quem tem participação obrigatória na Daily é o time de desenvolvimento apenas, o PO e o SM são apenas opcional sua participação. Na retrospectiva tem participação obrigatória o time de desenvolvimento e o SM.

O Refinamento acontece um passo antes da Planning (no final da sprint anterior)

O principal objetivo da Review é verificar se a demanda agrega valor ao negócio