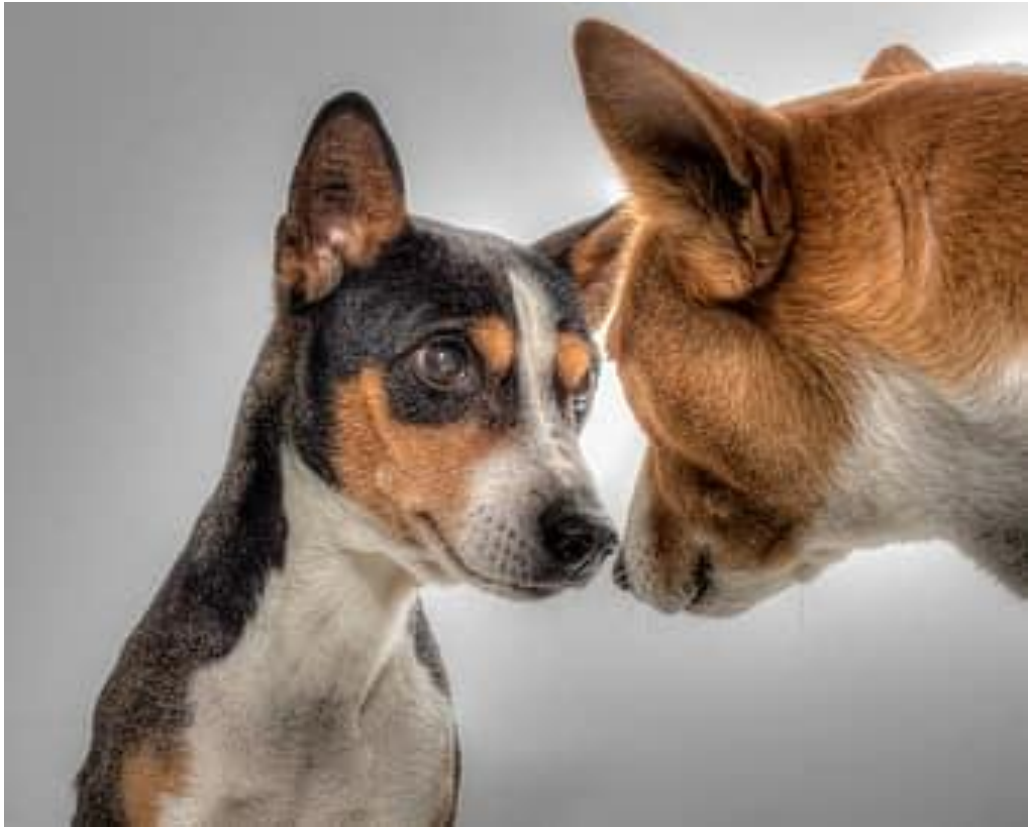
Metodologia de Avaliação Semanal do Projeto

Flavius Gorgônio
flavius.gorgonio@ufrn.br



Por que isso é importante?

O projeto é o método principal de avaliação da disciplina

Avalia-se, particularmente, a curva de aprendizado e não o resultado final

O processo de avaliação é contínuo, assim, todo dia é dia de avaliação

Aproxima-se da forma como o indivíduo é avaliado dentro das empresas e instituições

Critérios de avaliação

- Compromisso e pontualidade nas entregas
  - O cumprimento dos prazos é um compromisso do desenvolvedor
  - **Não** haverá prorrogação nos prazos de entrega, mesmo por problemas técnicos
- Clareza e padronização do código fonte
  - Martin, R. C. (2019). Código limpo: habilidades práticas do Agile software. Alta Books. Disponível em: https://bit.ly/34vcnMO
- Criatividade e inovação no código produzido
  - Inspiração nos princípios do *design thinking*
  - Brown, T. (2020). Design Thinking: uma metodologia poderosa para decretar o fim das velhas ideias. Alta Books. Disponível em: https://bit.ly/3h2PepZ
- Eficácia e eficiência no código produzido
  - Testes, identificação e eliminação de erros, refatoração

Pontuação do projeto

Critérios	Nota Máxima
➢ O aluno ou aluna não postou o link da repositório na tarefa do SIGAA ➢ O repositório informado está errado ou não existe ➢ O código fonte do programa não foi disponibilizado ➢ O código fonte não compila ➢ O código fonte apresentado é plágio no todo ou em parte	0.0
➢ O código fonte é apenas uma versão alterada do modelo exemplo	3.0
➢ O código fonte não implementa nem o mínimo do que foi solicitado	4.0
➢ O código fonte implementa APENAS o que foi solicitado	5.0
➢ O código fonte implementa MAIS do que foi solicitado	7.0
➢ O código fonte implementa MUITO MAIS do que foi solicitado	10.0

Desculpas mirabolantes que não serão aceitas...

- “Professor, faltou energia/internet/água/gás na minha casa e não consegui enviar o projeto na data prevista”
  Isso acontece com quem deixa as coisas para a última hora, da próxima vez programe-se com antecedência para evitar eventuais problemas.

- “Professor, era semana de provas, não tem só a sua disciplina não...”
  Bem vindo ao mundo real, da próxima vez não acumule atividades.

- “Ah professor, eu vi um exemplo ‘parecido’ na internet e me baseei nele, nem sei mais onde foi que vi...”
  Isso é plágio! E plágio é crime! Da próxima vez lembre-se de incluir os devidos créditos em cada trecho de código que não foi desenvolvido por você.

Desculpas mirabolantes que não serão aceitas...

- “Eita, essa parte aí do projeto foi Fulaninho quem fez, tem que perguntar a ele”
  O projeto é de responsabilidade de toda a equipe, isso só demonstra que você NÃO está integrado à equipe

- “Espera, professor, faz tanto tempo que fiz isso que nem lembro...”
  Programação não se “esquece”, não é uma questão de memorizar: ou se sabe ou não se sabe!

- “Prof, não houve aula, então a entrega do projeto será adiada?”
  Não, o projeto segue um cronograma próprio, você tem entregas a fazer toda semana, independente de haver ou não haver aula.

Desculpas mirabolantes que não serão aceitas...

- “Professor, minha tataravó morreu, mamãe ficou muito abatida, meu cachorro está doente, choveu, tinha uma goteira em cima da minha mesa e molhou o computador, eu perdi o ônibus e o pneu da minha bicicleta amanheceu furado... Será que não daria para adiar a entrega???”
  Não! O projeto é uma atividade de desenvolvimento contínuo, seja precavid@ e esteja preparado para eventuais imprevistos e eventos desagradáveis !

- “Professor, o senhor não tem coração?”
  Tenho, mas tomo todas as minhas decisões com o cérebro!

- “Professor, se eu ficar reprovado minha mãe me mata!”
  Nada, mãe sempre perdoa... mas em uma empresa você estaria demitid@!

Semana 1: Identificação e apresentação do projeto

O objetivo da Semana 1 é apresentar o tema do projeto. Crie algumas telas com informações básicas sobre o seu projeto e disponibilize-o no Github. O código que exibe essas telas deve estar em uma função diferente da função main() e deve ser chamada a partir desta. Veja um exemplo em: https://github.com/FlaviusGorgonio/LinguaSolta.git.

O código apresentado no exemplo corresponde ao mínimo necessário e serve apenas como um roteiro para que @s alun@s saibam o que deve ser feito. Sintam-se à vontade para modificar este código e incluir novas funcionalidades.

Tarefas a serem realizadas:

- ➢ Criar uma conta no Github
- ➢ Escolher o tema e a equipe (se houver) do seu projeto
- ➢ Configurar o seu ambiente de trabalho
- ➢ Produzir a versão 0.1 do seu projeto e disponibilizar no seu repositório no Github
- ➢ Enviar o link do seu repositório na atividade do SIGAA

Obs: só serão avaliados os projetos que forem submetidos pelo Github. Códigos enviados por e-mail, whatsapp ou outros meios não serão corrigidos. Atenção: Na sua resposta, envie APENAS o link do seu repositório!

Semana 2: Telas e interface do Módulo 1

O objetivo da Semana 2 é iniciar o projeto da interface. Crie todas as telas da interface referentes ao Módulo 1 do seu projeto, com menus, submenus e telas de entrada de dados. Cada tela deve ser representada em uma função diferente, chamadas e exibidas sequencialmente a partir da função main().

Veja código exemplo em: https://github.com/FlaviusGorgonio/LinguaSolta.git.

Obs: O código apresentado no exemplo corresponde ao mínimo necessário e serve apenas como um roteiro para que @s alun@s saibam que deve ser feito. Sintam-se à vontade para modificar este código e incluir novas funcionalidades.

Tarefas a serem realizadas:

- ➢ Criar todas as telas da interface
- ➢ Produzir a versão 0.2 do seu projeto e disponibilizar no seu repositório no Github
- ➢ Enviar o link do seu repositório na atividade do SIGAA

Obs: só serão avaliados os projetos que forem atualizados no Github e cujo link tenha sido disponibilizado pelo SIGAA.

Semana 3: Telas e interface dos Módulos 2 e 3

Crie as telas dos Módulos 2 e 3 da interface incluindo os menus e submenus do seu projeto. Cada tela deve ser representada em uma função diferente, que devem ser chamadas e exibidas sequencialmente a partir da função main(). Veja um exemplo em: https://github.com/FlaviusGorgonio/LinguaSolta.git.

Obs: O código apresentado no exemplo corresponde ao mínimo necessário e serve apenas como um roteiro para que @s alun@s saibam que deve ser feito. Sintam-se à vontade para modificar este código e incluir novas funcionalidades.

Tarefas a serem realizadas:

- ➢ Criar todas as telas da interface
- ➢ Produzir a versão 0.3 do seu projeto e disponibilizar no seu repositório no Github
- ➢ Enviar o link do seu repositório na atividade do SIGAA

Obs: só serão avaliados os projetos que forem submetidos pelo Github

Semana 4: Estrutura de navegação entre módulos

O objetivo da Semana 4 é criar uma estrutura de navegação de telas, na forma de menus e submenus, que integre as principais funcionalidades do sistema. Os submenus deverão possuir uma opção <<voltar>> para retornar ao menu anterior e o menu principal conter uma opção para encerrar do programa. Veja exemplo em: https://github.com/FlaviusGorgonio/LinguaSolta.git.

Atenção para o sistema de navegação implementado no projeto. Quando uma função é encerrada, o fluxo de execução volta automaticamente para a função chamadora.

Tarefas a serem realizadas:

- ➢ Interligar todas as telas da interface e permitir a navegação entre telas
- ➢ Produzir a versão 0.4 do seu projeto e disponibilizar no seu repositório no Github
- ➢ Enviar o link do seu repositório na atividade do SIGAA

Obs: só serão avaliados os projetos que forem atualizados no Github e cujo link tenha sido disponibilizado pelo SIGAA.

Semana 5: Modularização do programa

O objetivo da Semana 5 é dividir o seu projeto em arquivos, cada um contendo funcionalidades relacionadas a um determinado módulo do sistema. Os módulos devem ser o mais independentes possível, de forma que possam ser reutilizados em outros programas. Veja exemplo em: https://github.com/FlaviusGorgonio/LinguaSolta.git.

O código apresentado no exemplo corresponde ao mínimo necessário e serve apenas como um roteiro sobre o que deve ser feito. Sintam-se à vontade para criar outros módulos e incluir novas funcionalidades.

Tarefas a serem realizadas:

- ➢ Criar arquivos de módulos e importá-los a partir do programa principal
- ➢ Produzir a versão 0.5 do seu projeto e disponibilizar no seu repositório no Github
- ➢ Enviar o link do seu repositório na atividade do SIGAA

Obs: só serão avaliados os projetos que forem atualizados no Github e cujo link tenha sido disponibilizado pelo SIGAA.

Semana 6: Entrega do Módulo I - Interface

O objetivo da Semana 6 é entregar a primeira versão do seu projeto, que corresponde ao Módulo Interface. Nesta *release*, TODAS as telas do programa devem estar disponíveis para navegação e todas as funcionalidades do programa devem ser apresentadas como opções de menus. As funcionalidades que ainda não foram implantadas devem ser representadas com telas de aviso ou de simulação. Veja exemplo em: https://github.com/FlaviusGorgonio/LinguaSolta.git.

Atenção para o sistema de navegação implementado no projeto. Quando uma função é encerrada, o fluxo de execução volta automaticamente para a função chamadora.

Tarefas a serem realizadas:

- ➢ Possibilitar acesso a todas as telas da interface e funcionalidades previstas no projeto
- ➢ Produzir a versão 0.6 do seu projeto e disponibilizar no seu repositório no Github
- ➢ Enviar o link do seu repositório na atividade do SIGAA

Obs: só serão avaliados os projetos que forem atualizados no Github e cujo link tenha sido disponibilizado pelo SIGAA.

Semana 7: Validação dos dados de entrada

Desenvolva um conjunto de funções de validação para validar TODOS os dados informados pelo usuário. Tanto os campos de entrada de dados quanto as opções de menu nas telas de navegação devem ser validados. Veja exemplo em: https://github.com/FlaviusGorgonio/LinguaSolta.git.

Campos de dados devem ser verificados quanto ao tamanho e conteúdo. Por exemplo, é necessário validar datas, nomes, endereços postais e de email, números de telefone, CPF, RG, CNPJ, etc.

Nenhuma informação deve ser armazenada incorretamente se for possível verificá-la.

Tarefas a serem realizadas:

- ➢ Validar todas as entradas de dados
- ➢ Validar o conteúdo dos campos de informação do usuário
- ➢ Produzir a versão 0.7 do seu projeto e disponibilizar no seu repositório no Github
- ➢ Enviar o link do seu repositório na atividade do SIGAA

Obs: só serão avaliados os projetos que forem atualizados no Github e cujo link tenha sido disponibilizado pelo SIGAA.

Semana 8: Biblioteca de funções de validação

Crie uma biblioteca com funções para validação desenvolvidas anteriormente. As funções devem ser genéricas o bastante para que possam ser reutilizadas em outros programas. Utilize as técnicas de compilação em separado para adicionar a biblioteca ao seu projeto. Veja exemplo em: https://github.com/FlaviusGorgonio/LinguaSolta.git.

ATENÇÃO: As funções apresentadas no projeto exemplo servem apenas para orientar as equipes sobre como criar suas próprias funções. Copiar código sem incluir referência é PLÁGIO, passível de punição, caso utilize códigos de outr@s programador@s, não esqueça de mencionar.

Tarefas a serem realizadas:

- ➢ Criar (ou adaptar) funções que possam ser utilizadas para validação de dados de entrada digitados pelo usuário
- ➢ Produzir a versão 0.8 do seu projeto e disponibilizar no seu repositório no Github
- ➢ Enviar o link do seu repositório na atividade do SIGAA

Obs: só serão avaliados os projetos que forem atualizados no Github e cujo link tenha sido disponibilizado pelo SIGAA.

Semana 9: Tipos abstratos de dados

Crie tipos abstratos de dados para o seu projeto que possibilitem agregar os dados coletados de uma mesma entidade em uma estrutura (ou união). Analise e dimensione os tamanhos dos campos de forma adequada para evitar desperdícios de memória.

Inclua esses tipos de dados nos arquivos de cabeçalho dos seus respectivos módulos do sistema. Os módulos devem possuir o máximo de independência entre si, de forma que possam ser reutilizados em outros programas. Veja exemplo em: https://github.com/FlaviusGorgonio/LinguaSolta.git.

Tarefas a serem realizadas:

- ➢ Criar tipos de dados abstratos nos diversos módulos do seu projeto, importando-os para os demais módulos que os utilize
- ➢ Produzir a versão 0.9 do seu projeto e disponibilizar no seu repositório no Github
- ➢ Enviar o link do seu repositório na atividade do SIGAA

Obs: só serão avaliados os projetos que forem atualizados no Github e cujo link tenha sido disponibilizado pelo SIGAA.

Semana 10: Armazenamento de dados em arquivo

Adicione arquivos (do tipo texto e/ou binário) ao seu projeto para armazenamento definitivo dos dados cadastrados pelo usuário. A escolha entre arquivos do tipo texto/binários dependerá do tipo de projeto e da natureza dos dados armazenados. Porém, independente do tipo de projeto, você deverá armazenar os dados em arquivos de forma permanente.

Cada arquivo deverá conter os dados e informações cadastradas em cada módulo e o programa deverá ser capaz de recuperar os dados previamente cadastrados em uma execução anterior. Veja exemplo em: https://github.com/FlaviusGorgonio/LinguaSolta.git.

Tarefas a serem realizadas:

- ➢ Armazenar os dados cadastrados pelo usuário em arquivos do tipo binário e escrever funções que permitam a recuperação dos dados armazenados
- ➢ Produzir a versão 0.10 do seu projeto e disponibilizar no seu repositório no Github
- ➢ Enviar o link do seu repositório na atividade do SIGAA

Obs: só serão avaliados os projetos que forem atualizados no Github e cujo link tenha sido disponibilizado pelo SIGAA.

Semana 11: Exclusão lógica e física de dados

Implemente módulos que permitam realizar tanto a exclusão lógica quanto a exclusão física dos dados existentes nos arquivos do tipo binário existentes no seu projeto.

A exclusão lógica deverá ocorrer quando o usuário optar por excluir um dado que tenha sido anteriormente armazenado. A exclusão física será executada somente quando o usuário desejar apagar TODOS os dados do programa.

Cada estrutura deverá conter um campo status que indica se aquele registro foi ou não excluído (logicamente). O programa deverá ainda ser capaz de recuperar os dados excluídos (logicamente) em uma execução anterior. Veja exemplo em: https://github.com/FlaviusGorgonio/LinguaSolta.git.

Tarefas a serem realizadas:

- ➢ Organizar o código do seu projeto em módulos consistentes, eliminando trechos obsoletos e/ou desnecessários
- ➢ Produzir a versão 0.11 do seu projeto e disponibilizar no seu repositório no Github
- ➢ Enviar o link do seu repositório na atividade do SIGAA

Obs: só serão avaliados os projetos que forem atualizados no Github e cujo link tenha sido disponibilizado pelo SIGAA.

Semana 12: Entrega do Módulo II - Dados

Finalize a segunda versão do seu projeto, que corresponde ao Módulo Dados. Nesta *release*, TODOS os dados informados pelo usuário devem ser armazenados de forma definitiva pelo sistema.

Os arquivos (texto e/ou binário) deverão conter os dados cadastrados pelo usuário e o programa deverá ser capaz de recuperar os dados previamente cadastrados em uma execução anterior através de consultas e relatórios.

Veja exemplo em:
https://github.com/FlaviusGorgonio/LinguaSolta.git.

Tarefas a serem realizadas:

- ➢ Armazenar os dados cadastrados pelo usuário em arquivos do tipo texto e escrever funções que permitam a recuperação dos dados armazenados
- ➢ Produzir a versão 0.12 do seu projeto e disponibilizar no seu repositório no Github
- ➢ Enviar o link do seu repositório na atividade do SIGAA

Obs: só serão avaliados os projetos que forem atualizados no Github e cujo link tenha sido disponibilizado pelo SIGAA.

Semana 13: Relatórios de listagem de dados

Implemente relatórios que possibilitem a listagem do conteúdo que foi armazenado em cada um dos arquivos. Os relatórios podem exibir todos os dados do arquivo ou apenas uma parte deles, ou seja, apenas alguns campos de todos os registros. Os dados devem ser apresentados na forma de uma listagem simples, na mesma ordem em que forma cadastrados.

Veja exemplo: https://github.com/FlaviusGorgonio/LinguaSolta.git.

Lembre-se: ***compromisso*** e ***pontualidade*** nas entregas também são importantes critérios de avaliação.

Tarefas a serem realizadas:

- ➢ Criar relatórios que listem os dados armazenados em cada um dos arquivos do seu projeto
- ➢ Produzir a versão 0.13 do seu projeto e disponibilizar no seu repositório no Github
- ➢ Enviar o link do seu repositório na atividade do SIGAA

Obs: só serão avaliados os projetos que forem atualizados no Github e cujo link tenha sido disponibilizado pelo SIGAA.

Semana 14: Relatórios de listagem com filtros

Implemente relatórios que permitam filtrar parte dos registros cadastrados nos arquivos a partir de campos específicos definidos pelo usuário. Por exemplo, você pode filtrar alunos por curso, clientes por cidade ou estado, produtos por marca, vendas por período de data inicial e final, etc.

Seu programa deve ter funções que recebem um ou mais campos como parâmetro e exibem os registros que atendem aos critérios indicados nesses campos.

Os relatórios devem exibir os dados na forma de uma listagem, na mesma ordem em que estes foram cadastrados anteriormente. Veja exemplo: https://github.com/FlaviusGorgonio/LinguaSolta.git.

Tarefas a serem realizadas:

- ➢ Criar relatórios com filtros a partir de um (ou mais) campos da estrutura de dados utilizada
- ➢ Produzir a versão 0.14 do seu projeto e disponibilizar no seu repositório no Github
- ➢ Enviar o link do seu repositório na atividade do SIGAA

Obs: só serão avaliados os projetos que forem atualizados no Github e cujo link tenha sido disponibilizado pelo SIGAA.

Semana 15: Relatórios de várias fontes de dados

Implemente relatórios que permitam combinar dados provenientes de mais de um arquivo.

Por exemplo, suponha que você registrou uma venda armazenando a data da compra, o CPF do cliente e o código do produto adquirido. Agora, você deseja listar todas as compras de um determinado cliente a partir do seu CPF, mas quer exibir o nome do cliente e o nome do produto adquirido, obtendo essas informações de outros arquivos.

Os relatórios devem exibir os dados na forma de uma listagem simples, na mesma ordem em que forma cadastrados. Veja exemplo: https://github.com/FlaviusGorgonio/LinguaSolta.git.

Tarefas a serem realizadas:

- ➢ Criar relatórios que listem os dados armazenados em cada um dos arquivos do seu projeto
- ➢ Produzir a versão 0.15 do seu projeto e disponibilizar no seu repositório no Github
- ➢ Enviar o link do seu repositório na atividade do SIGAA

Obs: só serão avaliados os projetos que forem atualizados no Github e cujo link tenha sido disponibilizado pelo SIGAA.

Semana 16: Relatórios com listas dinâmica

Implemente relatórios que recuperem os dados cadastrados nos arquivos e organize-os em uma lista dinâmica.

Por exemplo, você pode recuperar os alunos cadastrados no arquivo e montar uma lista dinâmica apenas com aqueles que atendem a um determinado critério (filtro).

Os relatórios devem exibir os dados na forma de uma listagem. Veja exemplo: https://github.com/FlaviusGorgonio/LinguaSolta.git.

Tarefas a serem realizadas:

- ➢ Criar relatórios dinâmicos, ordenando os registros a partir de um dos campos da estrutura de dados utilizada
- ➢ Produzir a versão 0.16 do seu projeto e disponibilizar no seu repositório no Github
- ➢ Enviar o link do seu repositório na atividade do SIGAA

Obs: só serão avaliados os projetos que forem atualizados no Github e cujo link tenha sido disponibilizado pelo SIGAA.

Semana 17: Relatórios com critérios de ordenação

Implemente relatórios que recupere os dados cadastrados nos arquivos e organize-os em uma lista dinâmica, numa ordem diferente daquela em que os dados estão dispostos originalmente.

Por exemplo, você pode recuperar os alunos cadastrados no arquivo, em qualquer ordem, e exibi-los em ordem alfabética (ou qualquer outro campo que o usuário deseje). Ou então, pode exibir os professores ordenados por data de aniversário.

Os relatórios devem exibir os dados na forma de uma listagem. Veja exemplo: https://github.com/FlaviusGorgonio/LinguaSolta.git.

Tarefas a serem realizadas:

- ➢ Criar relatórios dinâmicos, ordenando os registros a partir de um dos campos da estrutura de dados utilizada
- ➢ Produzir a versão 0.17 do seu projeto e disponibilizar no seu repositório no Github
- ➢ Enviar o link do seu repositório na atividade do SIGAA

Obs: só serão avaliados os projetos que forem atualizados no Github e cujo link tenha sido disponibilizado pelo SIGAA.

Semana 18: Entrega do Módulo III - Projeto Final

Finalize a terceira e última versão do seu projeto, que corresponde ao Módulo Relatórios. Nesta *release*, TODAS as funcionalidades do sistema devem estar implementadas de forma definitiva.

Os arquivos (texto e/ou binário) deverão conter os dados cadastrados pelo usuário e o programa deverá ser capaz de recuperar os dados previamente cadastrados em uma execução anterior através de consultas e relatórios.

O sistema deve possuir relatórios dinâmicos, criados a partir de estruturas com alocação dinâmica de memória. Veja exemplo em: https://github.com/FlaviusGorgonio/LinguaSolta.git.

Tarefas a serem realizadas:

- ➢ Armazenar os dados cadastrados pelo usuário em arquivos do tipo texto e escrever funções que permitam a recuperação dos dados armazenados
- ➢ Produzir a versão 1.0 do seu projeto e disponibilizar no seu repositório no Github
- ➢ Enviar o link do seu repositório na atividade do SIGAA

Obs: só serão avaliados os projetos que forem atualizados no Github e cujo link tenha sido disponibilizado pelo SIGAA.