

UNIVERSIDADE FEDERAL DO PARANÁ

NOME DO(A) ALUNO(A) NIVIADOMSKI · GRR20195142

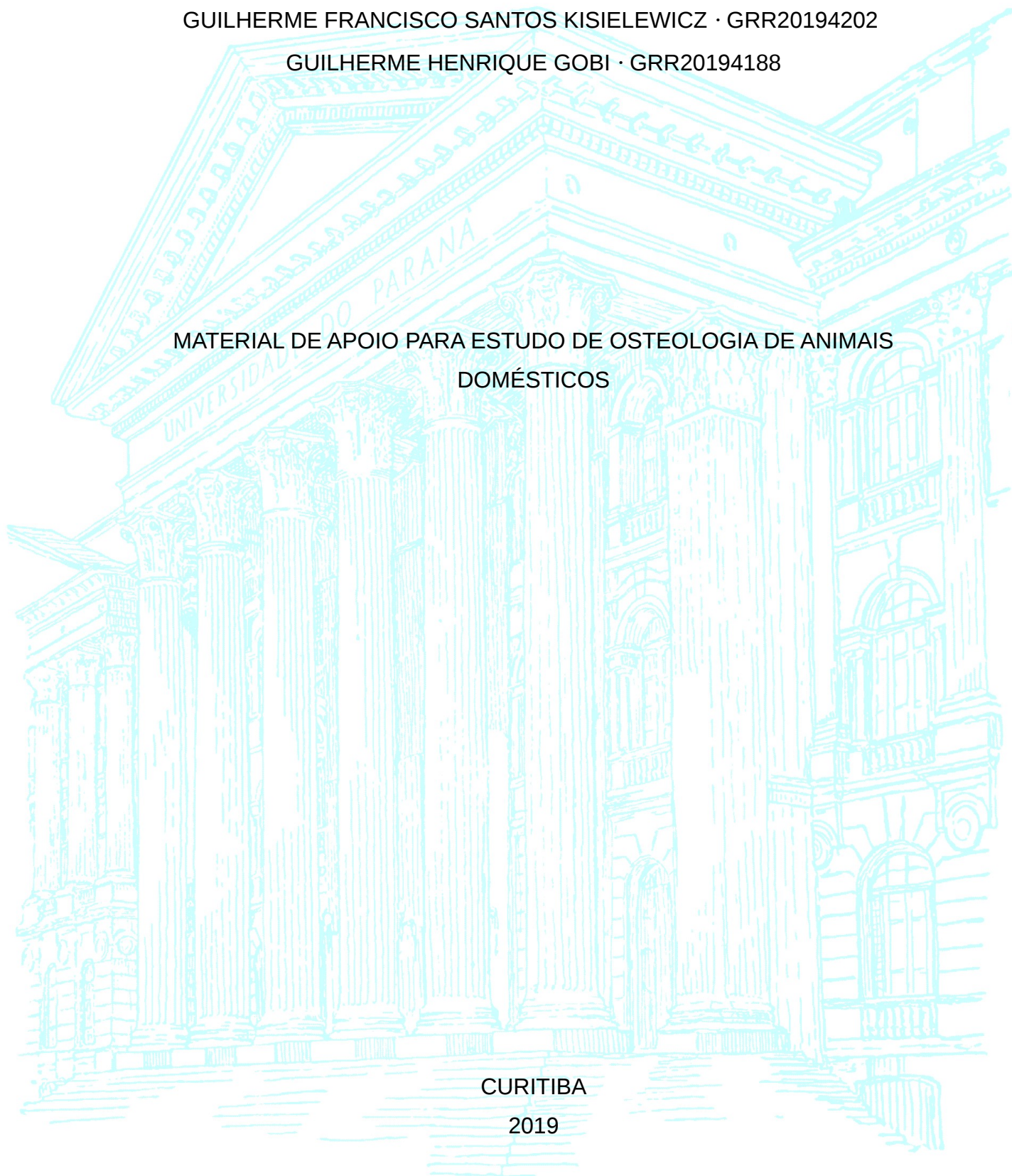
GUILHERME FRANCISCO SANTOS KISIELEWICZ · GRR20194202

GUILHERME HENRIQUE GOBI · GRR20194188

MATERIAL DE APOIO PARA ESTUDO DE OSTEOLOGIA DE ANIMAIS  
DOMÉSTICOS

CURITIBA

2019



NOME DO(A) ALUNO(A) NIVIADOMSKI · GRR20195142  
GUILHERME FRANCISCO SANTOS KISIELEWICZ · GRR20194202  
GUILHERME HENRIQUE GOBI · GRR20194188

MATERIAL DE APOIO PARA ESTUDO DE OSTEOLOGIA DE ANIMAIS  
DOMÉSTICOS

Relatório apresentado ao curso de graduação em Engenharia Agrônoma, Setor de Ciências Agrárias, Universidade Federal do Paraná, como requisito parcial à obtenção da conclusão da disciplina de Fundamentos de Programação de Computadores (Python).

CURITIBA  
2019

## RESUMO

O presente trabalho acadêmico foi elaborado para a disciplina de Fundamentos de Programação de Computadores em conjunto a disciplina de Anatomia dos Animais Domésticos e irá apresentar um novo método de estudo que pode ser aplicado a qualquer curso que tenha a disciplina com conteúdo similar. Iremos apresentar um simulador de conhecimentos que traz uma sequência de imagens obtidas diretamente na universidade e fornece uma pontuação de acordo com seus acertos sobre osteologia de animais domésticos. Esse meio de estudo também é um facilitador para estudos remotos as aulas laboratoriais, com materiais confiáveis, oficiais para a execução de provas da universidade, adequados especialmente ao curso de Agronomia o qual possui apenas 60 horas semestrais e apenas 30 destas horas são aplicadas em práticas laboratoriais e com a ferramenta serão disponibilizados acessos ilimitados. Este software já estava em cogitação dentro do departamento de biológica e utilizamos dos conhecimentos adquiridos durante o semestre para desenvolver uma base teste para eles.

Palavras-chave: Python 1. Osteologia 2. Anatomia 3. Animal 4. Pillow 5. Kyvy 6.

## SUMÁRIO

1 INTRODUÇÃO.....	05
1.1 JUSTIFICATIVA.....	1
1.2 MOTIVAÇÃO.....	16
1.3 OBJETIVOS.....	16
1.4 FUNDAMENTAÇÃO TEÓRICA.....	16
2 METODOLOGIA.....	06
2.1 BIBLIOTECAS.....	
2.2 FUNÇÕES.....	
3 CONSIDERAÇÕES FINAIS.....	19
REFERÊNCIAS.....	20

## 1 INTRODUÇÃO

### 1.1 JUSTIFICATIVA

Em contato com o departamento de Anatomia fomos informados de um projeto que buscava levar materiais de apoio práticos a fácil acesso. Dentro desse projeto não havia ainda alguém responsável para o desenvolvimento de um software para concretizar a implementação de um módulo básico.

### 1.2 MOTIVAÇÃO

Em conjunto ao cronograma de provas da disciplina de Anatomia, pensamos em desenvolver um método que facilitasse o estudo de estruturas ósseas de animais domésticos. Visando o estudo de forma prática e rápida, com respostas imediatas de verificação de conhecimento do discente que utilizar a ferramenta.

### 1.3 OBJETIVOS

Mapear e integrar a maior quantidade de estruturas ósseas possíveis de equinos e bovinos, principalmente, e com maior quantidade de estruturas externas.

O desenvolvimento deste programa também busca auxiliar e acrescentar aulas laboratoriais e direcionadas à prática e identificação presente das partículas.

### 1.4 FUNDAMENTAÇÃO TEÓRICA

De acordo com o dicionário Michaelis (2019), osteologia é o “Estudo da estrutura, da forma, da natureza e do desenvolvimento dos ossos.” Em suma, aprendemos ao longo do curso da disciplina que osteologia é o estudo de ossos, cartilagens e seus anexos.

## 2 METODOLOGIA

### 2.1 BIBLIOTECAS

As bibliotecas utilizadas para execução do programa são os módulos PIL e kivy 1.11.

Python Imaging Library (PIL) é uma biblioteca da linguagem de programação Python que adiciona suporte à abertura e gravação de muitos formatos de imagem diferentes, serve para alterar as imagens como rotacionar, mudar a cor, aumentar e/ou diminuir e converter formatos de imagem. E Kivy é um módulo de interface gráfica, uma biblioteca open source escrita em Python para o desenvolvimento de aplicativos móveis ou desktop.

### 2.2 FUNÇÕES

```
File Edit Format Run Options Window Help
from os import listdir, path
from os.path import isfile, join
from random import randint
import difflib
from PIL import Image
import ast
import shutil
from kivy,app import App
from kivy.uix.screenmanager import Screen

class JanelaPrincipal(Screen): # É as informações dos elementos da interface do arquivo .kv
    pass

def self(args): # chama as outras funções
    pass

class Janela(App): # classe onde são associadas todas as funções da interface

    config = [f for f in listdir("data/") if isfile(join("data/", f))] # É quando arquivos possui a pasta "data/" e junta "data/" ao nome deles, para armazenar os arquivos corretamente por outras funções
    peca=config[randint(0,len(config)-1)] # seleciona aleatoriamente um arquivo de configuração

    def build(self): # constrói os elementos da janela segundo as configurações do arquivo .kv
        return JanelaPrincipal()

    def randomize(self): # escolhe aleatoriamente uma imagem e os pontos dela
        x=self.config[randint(0,(len(self.config)-1))] # escolhe um arquivo aleatoriamente
        abrir=open("data/"+x) # abre o arquivo escolhido
        linha=abrir.readlines() # transforma o arquivo em uma lista de que cada linha é um item
        imagempeca=linha[2] # seleciona a terceira linha, onde está o endereço da imagem
        imagempeca=imagempeca[7:] # remove o identificador da linha
        imagempeca=imagempeca[:-1] # remove o indicador de nova linha
        pontos=linha[5:] # seleciona os pontos
        pontos=pontos[randint(0,(len(pontos)-1))] # pega um ponto aleatório
        pontos=ast.literal_eval(pontos) # converte cada item dentro do ponto para os seus valores de integrais e strings
        img=Image.open(imagempeca) # abre a imagem selecionada
        seta=Image.open("data/images/arrow.png") # abre a imagem de seta
        img.paste(seta,pontos[:2],seta) # cola a seta nos coordenadas indicadas pelo ponto
```

File Edit Format Run Options Window Help

```

def randomize(self): # escolhe aleatoriamente uma imagem e os pontos dela
    x=self.configs[randint(0,(len(self.configs)-1))] # escolhe um arquivo aleatoriamente
    abrir =open("data/"+x) # abre o arquivo escolhido
    linha=abrir.readlines() # transforma o arquivo em uma lista em que cada linha é um item
    imagempeca=linha[2] # seleciona a terceira linha, onde está o endereço da imagem
    imagempeca=imagempeca[7:] # remove o identificador da linha
    imagempeca=imagempeca[:-1] # remove o anteceder de nova linha
    pontos=linha[5:] # seleciona os pontos
    pontos=pontos[randint(0,(len(pontos)-1))] # pega um ponto aleatório
    pontos=ast.literal_eval(pontos) # converte cada item dentro do pontos para os seus valores de inteiros e strings
    img=Image.open(imagempeca) # abre a imagem selecionada
    seta=Image.open("data/images/arrow.png") # abre a imagem da seta
    img.paste(seta,pontos[2],seta) # colar a seta nas coordenadas indicadas pelo ponto
    img.save("data/cache/output.png") # salva a imagem
    nome=pontos[2] # pega o nome do ponto
    nomecache=open("data/cache/nomecache.txt","wb") # abre o arquivo temporário do nome
    nomecache.write(nome.encode('utf-8','ignore')) # salva o nome do ponto para um arquivo temporário
    nomecache.close() # fecha o arquivo

def end(self):
    print(self.ptn)

def resq(self, text): # compara o input do usuário com o gerado e retorna a sua pontuação
    score = open("data/cache/score.txt", "r") # abre o arquivo de pontuação
    pontuacao = score.readlines() # separa as linhas do arquivo
    pontuacao = float(pontuacao[0]) # seleciona a primeira linha
    score.close() # fecha o arquivo de pontuação
    nome=open("data/cache/nomecache.txt") # abre o arquivo com o nome do ponto
    nomecache=nome.readline() # lê o arquivo do nome
    nome.close() # fecha o arquivo do nome

    comparacao=diffib.SequenceMatcher(None,text,nomecache).ratio() # faz a comparação entre o input do usuário e o nome do ponto

    if comparacao==1: # se o nome for igual

```

Ln 46 Col 9

File Edit Format Run Options Window Help

```

def resq(self, text): # compara o input do usuário com o gerado e retorna a sua pontuação
    score = open("data/cache/score.txt", "r") # abre o arquivo de pontuação
    pontuacao = score.readlines() # separa as linhas do arquivo
    pontuacao = float(pontuacao[0]) # seleciona a primeira linha
    score.close() # fecha o arquivo de pontuação
    nome=open("data/cache/nomecache.txt") # abre o arquivo com o nome do ponto
    nomecache=nome.readline() # lê o arquivo do nome
    nome.close() # fecha o arquivo do nome

    comparacao=diffib.SequenceMatcher(None,text,nomecache).ratio() # faz a comparação entre o input do usuário e o nome do ponto

    if comparacao==1: # se o nome for igual
        pontuacao=pontuacao+1.0 # aumenta a pontuação
        pontuacao=str(pontuacao) # converte a pontuação em string
        f=open("data/cache/score.txt","wb") # abre o arquivo de pontuação
        f.write(pontuacao.encode('utf-8','ignore')) # escreve a nova pontuação
        f.close() # fecha o arquivo

    elif comparacao >=0.75 and comparacao < 1: # se o input do usuário não for exato
        pontuacao = pontuacao + 0.8 # atribui um valor parcial
        pontuacao = str(pontuacao)
        f = open("data/cache/score.txt", "wb")
        f.write(pontuacao.encode('utf-8', 'ignore'))
        f.close()

    elif comparacao >=0.6 and comparacao < 0.75:
        pontuacao = pontuacao + 0.6
        pontuacao = str(pontuacao)
        f = open("data/cache/score.txt", "wb")
        f.write(pontuacao.encode('utf-8', 'ignore'))
        f.close()

    else: # se o input for muito diferente
        pontuacao = pontuacao + 0.0 # não aumenta a pontuação
        pontuacao = str(pontuacao)

```

Ln 46 Col 9



```

File Edit Format Run Options Window Help
f.write(pontuacao.encode('utf-8', 'ignore'))
f.close()

else: # se o input for muito diferente
    pontuacao = pontuacao + 0.0 #não aumenta a pontuação
    pontuacao = str(pontuacao)
    f = open("data/cache/score.txt", "a")
    f.write(pontuacao.encode('utf-8', 'ignore'))
    f.close()

log = "\nResposta: " + text + " / gabarito: " + nomecache + " / pontuação: " + str(pontuacao) #cria uma string comparando o input com o gabarito e a pontuação atual

hist = open("data/cache/historico.txt", "a") #abre o arquivo de log temporário da sessão
hist.write(log.encode('utf-8', 'ignore')) #escreve a string de log no arquivo
hist.close() #fecha o arquivo

def pontuacao(self): # Função para mostrar a pontuação na tela

    ptm = open("data/cache/score.txt", "r") #abre o arquivo de pontuação
    self.pontuacao = ptm.readlines() #le a pontuação escrita nele
    self.pontuacao = self.pontuacao[0] #retorna a linha
    return self.pontuacao #devolve o valor da linha

ptm.pontuacao(self) #disponibiliza o valor da pontuação para acesso da classe

def resetar(self): #cria um log de última etapa e deleta os arquivos temporários

    his=path.realpath("data/cache/historico.txt") #é o endereço absoluto do arquivo
    log=his+".log" #coloca a extensão .log ao arquivo
    shutil.copy(his,log) #cria um arquivo de log definitivo
    open("data/cache/historico.txt","w").close() #limpa o arquivo historico
    reset=open("data/cache/score.txt","w") #abre o arquivo score
    reset.write("0.0") #limpa o arquivo de score
    reset.close() #fecha o arquivo

if __name__ == '__main__':
    janela().run() #liga principal da interface grafica

```

### 3 CONSIDERAÇÕES FINAIS

Com o desenvolvimento do programa apresentado nesse documento, utilizando a plataforma Python para desenvolvimento do software, obtiveram-se resultados esperados e eficientes para a prática de estudo do conteúdo de osteologia na anatomia. Os principais desafios foram as interações entre as funções para funcionarem em conjunto e o desenvolvimento da interface e sua funcionalidade. No mais, somos gratos pela oportunidade dada pelo Prof. MSc Jackson Antonio do Prado Lima para desenvolvermos esse conteúdo e iniciarmos nosso entendimento e desenvolvimento pelas pesquisas propostas pela Universidade.



## REFERÊNCIAS

MICHAELIS. Dicionário Brasileiro da Língua Portuguesa. Disponível em: <<https://michaelis.uol.com.br/moderno-portugues/busca/portugues-brasileiro/osteologia/>>. Acesso em: 18 jun. 2019

KYVY. Biblioteca Python. Disponível em: <<https://kivy.org/#home>>. Acesso em: 19 jun. 2019

JUNIOR, Amilton da Rocha Leal. Anatomia dos Animais Domésticos para Agronomia. 2019