Руководство разработчика к программе обработки базы данных ракет носителей

Авторы:

Животов Глеб

Миронюк Даниил

Игуменова Марта

1. Структура исходных данных

Исходные данные должны быть сохранены в двоичном файле, сформированном с помощью модуля pickle и имеющем название «Rick.pic». Данный файл должен храниться в подкаталоге «Data». Двоичный файл должен хранить в себе словарь словарей, ключами которого являются натуральные числа без повторений. Вложенные словари имеют следующие ключи в соответствующем порядке:

|  |  |  |
| --- | --- | --- |
| Ключ | Тип данных | Описание |
| name | String | Название ракеты-носителя |
| year | Integer | Год первого запуска |
| dev | String | Страна или объединение стран-разработчиков ракеты-носителя |
| mass | Float | Масса собственно ракеты в тоннах |
| efmass | Float | Масса полезной нагрузки, выводимой на орбиту в килограммах |
| stages | Integer | Количество ступеней |

Так же в рамках программы производится вычисление таких параметров, как дисперсия, отклонения и средние значения определённых выборок.

Так же в подкаталоге «Scripts» находится текстовый файл с названием «colors.txt», содержащий цвета для интерфейса, название и размер шрифта.

2. Структура каталогов приложения

Все составные части программы хранятся в каталоге «Work», местоположение которого не играет роли, т.к. определяется автоматически в ходе работы программы. Однако, стоит выбирать правильную директорию при запуске программы средствами командной строки Windows.

Внутри каталога «Work» существует 6 подкаталогов:

***Data***<-база данных

***Graphics***<-графические элементы приложения

***Library***<-библиотека стандартных(универсальных) функций, разработанных бригадой, которые могут использоваться для создания других приложений

***Notes***<-Каталог для документации, в нём размещаются Руководства пользователя и разработчика

***Output***<-каталог текстового вывода приложения – отчётов о подведении итогов.

***Scripts***<-Каталог для хранения специализированных программных блоков, в частности файла с определением параметров приложения, файла с основным скриптом приложения.

3. Структура приложения (описание файлов – фрагменты и их значение)

Приложение состоит из трёх основных частей:

1. BD04.py – основной скрипт на языке Python3. Путём его запуска производится запуск программы и дальнейший вызов функций. Расположен в подкаталоге «Scripts»
2. ourmodule.py – модуль, разработанный бригадой и содержащий базовые функции для работы с файлами и базой данных
3. colors.txt – текстовый файл, содержащий значения четырёх цветов интерфейса, название шрифта и его размер

4. Стандартные функции разработчика

4.1 Функции модуля ourmodule

Следующие функции относятся к библиотеке стандартных функций

***Функция byte:***

"""

Функция считывания базы из двоичного файла

Автор: Животов Глеб

:param w: Словарь словарей базы данных

:param path1: Путь к двоичному файлу

:return: Словарь словарей базы данных

"""

***Функция update\_output:***

"""

Функция обновления окна вывода

Автор: Игуменова Марта

:param listbox: Объект класса Multibox, который будет обновлён

:param w: Словарь словарей который загрузится в listbox

:return: None

"""

***Функция summary:***

"""

Функция расчёта отклонений и записи результата в файл

Автор: Миронюк Даниил

:param f: Список - выборка записей, подходящих по параметру

:param askentry: Объект класса Entry, из которого будет считано имя файла

:return: None

"""

***Функция find\_bolsh:***

"""

Автор: Игуменова Марта

Функция поиска записей, значение в поле key которых больше num

:param w: Словарь словарей с данными базы

:param num: Значение столбца

:param key: Ключ столбца, в котором происходит поиск

:return: Список подходящих записей

"""

***Функция add\_node:***

"""

Функция добавления записи в словарь словарей

Автор: Животов Глеб

:param w: Словарь словарей базы данных

:param new\_name: Имя новой записи

:param new\_year: Год новой записи

:param new\_dev: Разработчик новой записи

:param new\_mass: Масса новой записи

:param new\_efmss: Эффективная масса новой записи

:param new\_stages: Количество ступеней новой записи

:return: Изменённый словарь словарей базы данных

"""

***Функция delete\_node:***

"""

Функция удаления записи

Автор: Игуменова Марта

:param w: Словарь словарей базы данных

:param key: Список полей записи, где 0 элемент - искомый ключ

:param multibox: Объект класса Multilistbox, который будет обновлён после удаления

:return: None

"""

4.2 Функции скрипта BD04.py

Функция colors:

"""

Функция считывания параметров интерфейса из файла

Автор: Животов Глеб

:param path8: Путь к текстовому файлу с параметрами

:param length: Количество цветов

:return: Список, состоящий из количества length цветов, названия шрифта и размера шрифта

"""

***Функция load\_base\_button:***

"""

Функция, вызываемая при нажатии кнопки загрузки. Вынужденное использование глобальной переменной из-за особенностей вызова функции при нажатии кнопки. Загружает базу данных из двоичного файла в оперативную память.

Автор: Игуменова Марта

:return: None

"""

***Функция open\_w\_summary:***

"""

Функция, вызываемая при нажатии кнопки подведения итого. Создаёт диалоговое с возможностью задания имени выходного файла

Автор: Миронюк Даниил

:param f: Список - выборка записей, подходящих по параметру

:param bgcolor1: Цвет фона 1

:param bgcolor3: Цвет фона 2

:param mainfont: Список - Название шрифта и размер

:return: None

"""

***Функция open\_w\_findmensh:***

"""

Функция, вызываемая при нажатии кнопки "Найти меньше". Создаёт окно для ввода интересующих данных и запуска функции поиска записей со значением выбранного поля меньше заданного

Автор: Животов Глеб

:param w: Словарь словарей базы данных

:param bgcolor1: Цвет фона 1

:param bgcolor2: Цвет фона 2

:param bgcolor3: Цвет фона 3

:param mainfont: Список - Название шрифта, размер

:return: None

"""

***Функция open\_w\_findbolsh:***

"""

Функция, вызываемая при нажатии кнопки "Найти больше". Создаёт окно для ввода интересующих данных и запуска функции

поиска записей со значением выбранного поля больше заданного

Автор: Миронюк Даниил

:param w: Словарь словарей базы данных

:param bgcolor1: Цвет фона 1

:param bgcolor2: Цвет фона 2

:param bgcolor3: Цвет фона 3

:param mainfont: Список - Название фона, размер фона

:return: None

"""

***Функция open\_w\_findbetween:***

"""

Функция, вызываемая при нажатии кнопки "Найти в диапазоне". Создаёт окно для ввода интересующих данных и запуска функции поиска записей со значением выбранного поля в заданном диапазоне

Автор: Игуменова Марта

:param w: Словарь словарей базы данных

:param bgcolor1: Цвет фона 1

:param bgcolor2: Цвет фона 2

:param bgcolor3: Цвет фона 3

:param mainfont: Список - Название шрифта, размер шрифта

:return: None

"""

***Функция open\_w\_findname:***

"""

Функция, вызываемая при нажатии кнопки "Поиск по имени". Создаёт окно для ввода интересующих данных и запуска функции поиска записей с заданным именем

Автор: Животов Глеб

:param w: Словарь словарей базы данных

:param bgcolor1: Цвет фона 1

:param bgcolor2: Цвет фона 2

:param bgcolor3: Цвет фона 3

:param mainfont: Список - Название шрифта, размер шрифта

:return: None

"""

***Функция click\_add\_node:***

"""

Функция, вызываемая при нажатии кнопки добавления новой записи. Считывает данные из полей и запускает функцию

добавления записи

Автор: Миронюк Даниил

:param w: Словарь словарей базы данных

:param namee: Объект класса Entry, содержащий новое имя

:param yeare: Объект класса Entry, содержащий новый год

:param deve: Объект класса Entry, содержащий нового разработчик

:param masse: Объект класса Entry, содержащий новую массу

:param efmasse: Объект класса Entry, содержащий новую эффективную массу

:param stagese: Объект класса Entry, содержащий новое количество степеней

:param multibox: Объект класса Multilistbox, который будет обновлён после добавления записи

:return: None

"""

***Функция open\_w\_save:***

"""

Автор: Игуменова Марта

Функция сохранения словаря словарей в двоичный файл

:param w: Словарь словарей базы данных

:param path1: Путь к двоичному файлу

:return: None

"""

5. Требуемые пакеты и библиотеки

tkinter – построение интерфейса

messagebox – подмодуль модуля tkinter отвечающий за вывод сообщений

pickle – работа с двоичными файлами

os, sys – работа с путями к файлам и модулям

ourmodule – базовые функции работы с базой данных

6. Технические требования

32 или 64 битная ОС, на которую возможна установка Python3(https://www.python.org)

Приложения

Приложение 1

Листинг модуля ourmodule

import pickle

from tkinter import \*

from tkinter import messagebox as mbox

import os

def byte(w, path1):

"""

Функци считывания базы из двоичного файла

Автор: Животов Глеб

:param w: Словарь словарей базы данных

:param path1: Путь к двоичному файлу

:return: Словарь словарей базы данных

"""

f = open(path1, "rb")

w = pickle.load(f)

f.close()

return w

def update\_output(listbox, w):

"""

Функция обновления окна вывода

Автор: Игуменова Марта

:param listbox: Объект класса Multibox, который будет обновлён

:param w: Словарь словарей который загрузится в listbox

:return: None

"""

listbox.delete(0, END)

for a in w.keys():

listbox.insert(END, (str(a), str(w[a]['name']), str(w[a]['year']), str(w[a]['dev']), str(w[a]['mass']),

str(w[a]['efmass']), str(w[a]['stages'])))

def summary(f, askentry):

"""

Функция расчёта отклонений и записи результата в файл

Автор: Миронюк Даниил

:param f: Список - выборка записей, подходящих по параметру

:param askentry: Объект класса Entry, из которого будет считано имя файла

:return: None

"""

outname = askentry.get()

if outname == '':

mbox.showerror("Отсутствует название", "Введите название файла")

return

keys = ["year", "mass", "efmass", "stages"]

average = {} # Словарь, содержащий средние значения клюей keys, его нужно вывести в файл вместе с записями

kvadr = {} # Словарь КВАДРАТОВ среднекважратичных отклонений от среднеарифметического, вывод в файл

disp = {} # Словарь дисперсии, вывод в файл

k = len(f)

if k == 0:

for key in keys:

average[key] = 0

kvadr[key] = 0

disp[key] = 0

elif k == 1:

for key in keys:

s = 0

kv = 0

for node in f:

s += float(node[key])

kv += float(node[key])\*float(node[key])

average[key] = float(s)/k

kvadr[key] = 0

disp[key] = 0

else:

for key in keys:

s = 0

kv = 0

for node in f:

s += float(node[key])

kv += float(node[key])\*float(node[key])

average[key] = float(s)/k

kvadr[key] = kv/k - average[key]\*average[key]

disp[key] = (kv - 2\*average[key]\*s + k\*average[key]\*average[key])/(k-1)

outname += '.txt'

path = os.getcwd()

n = path.find("\Scripts")

path1 = os.path.join(path[0:n] + "\Output", outname)

fileout = open(path1, "w")

for node in f:

print(node['name'], node['year'], node['dev'], node['mass'], node['efmass'], node['stages'], file=fileout)

print("Средние значения", file=fileout)

print("Год: ", average['year'], "Отклонение от среднего арифметического: ", kvadr['year']\*\*0.5,

"Дисперсия: ", disp['year'], file=fileout)

print("Масса: ", average['mass'], "Отклонение от среднего арифметического: ", kvadr['mass']\*\*0.5,

"Дисперсия: ", disp['mass'], file=fileout)

print("Эфективная масса: ", average['efmass'], "Отклонение от среднего арифметического: ", kvadr['efmass']\*\*0.5,

"Дисперсия: ", disp['efmass'], file=fileout)

print("Ступени: ", average['stages'], "Отклонение от среднего арифметического: ", kvadr['stages']\*\*0.5,

"Дисперсия: ", disp['stages'], file=fileout)

mbox.showinfo("Сохранено!", "Сохранение успешно")

def find\_mensh(w, key, num):

"""

Автор: Миронюк Даниил

Функция поиска записей, значение в поле key которых меньше num

:param w: Словарь словарей с данными базы

:param num: Значение столбца

:param key: Ключ столбца, в котором происходит поиск

:return: Список подходящих записей

"""

f = []

for a in w.keys():

if float(w[a][key]) <= num:

f.append(w[a])

return f

def find\_bolsh(w, key, num):

"""

Автор: Игуменова Марта

Функция поиска записей, значение в поле key которых больше num

:param w: Словарь словарей с данными базы

:param num: Значение столбца

:param key: Ключ столбца, в котором происходит поиск

:return: Список подходящих записей

"""

f = []

for a in w.keys():

if float(w[a][key]) >= num:

f.append(w[a])

return f

def find\_between(w, key, min, max):

"""

Автор: Животов Глеб

Функция поиска записей, значение в поле key которых больше min и меньше max

:param w: Словарь словарей с данными базы

:param min: минимальное значение столбца

:param max: максимальное значение столбца

:param key: Ключ столбца, в котором происходит поиск

:return: Список подходящих записей

"""

f = []

for a in w.keys():

if (float(w[a][key]) >= min) & (float(w[a][key]) <= max):

f.append(w[a])

return f

def find\_name(w,name):

"""

Автор: Миронюк Даниил

Функция поиска записей с именем, введённым пользователем

Получает на вход список словарей w

Возвращает список ключей списка w

"""

f = []

for i in w.keys():

if w[i]['name'] == name:

f.append(w[i])

return f

def add\_node(w,new\_name,new\_year,new\_dev,new\_mass,new\_efmss,new\_stages):

"""

Функция добавления записи в словарь словарей

Автор: Животов Глеб

:param w: Словарь словарей базы данных

:param new\_name: Имя новой записи

:param new\_year: Год новой записи

:param new\_dev: Разработчик новой записи

:param new\_mass: Масса новой записи

:param new\_efmss: Эффективная масса новой записи

:param new\_stages: Количество ступеней новой записи

:return: Изменённый словарь словарей базы данных

"""

i=0

nname = new\_name

nyear = new\_year

ndev = new\_dev

nmass = new\_mass

nefmass = new\_efmss

nstages = new\_stages

while str(i) in w.keys():

i += 1

w[str(i)] = {

"name": nname,

"year": nyear,

"dev": ndev,

"mass": nmass,

"efmass": nefmass,

"stages": nstages

}

return w

def change\_node(w, key, cname, cyear, cdev, cmass, cefmass, cstages):

"""

Функция изменения существующей записи

Автор: Миронюк Даниил

:param w: Словарь словарей базы данных

:param key: Ключ изменяемой записи

:param cname: Новое имя

:param cyear: Новый год

:param cdev: Новый разработчик

:param cmass: Новая масса

:param cefmass: Новая эффективная масса

:param cstages: Новое количество ступеней

:return: Изменённый словарь словарей

"""

w[str(key)]["name"] = cname

w[str(key)]["year"] = cyear

w[str(key)]["dev"] = cdev

w[str(key)]["mass"] = cmass

w[str(key)]["efmass"] = cefmass

w[str(key)]["stages"] = cstages

return w

def delete\_node(w, key, multibox):

"""

Функция удаления записи

Автор: Игуменова Марта

:param w: Словарь словарей базы данных

:param key: Список полей записи, где 0 элемент - искомый ключ

:param multibox: Объект класса Multilistbox, который будет обновлён после удаления

:return: None

"""

try:

key = key[0]

except IndexError:

mbox.showerror("Отсутсвует выделение", "Не выбрана строка для удаления. Выберите и попробуйте снова")

return

w.pop(str(key))

update\_output(multibox, w)

Приложение 2

Листинг скрипта BD04.py

from tkinter import \*

import pickle

import os

from tkinter import messagebox as mbox

import sys

path = os.getcwd()

n = path.find("\Scripts")

path8 = os.path.join(path[0:n] + "\Scripts", "colors.txt")

path1 = os.path.join(path[0:n] + "\Data", "Rick.pic") # ОП до рикпик чтобы прога работала везде куда ее скачают

path2 = os.path.join(path[0:n] + "\Graphics",

"fon.png") # ОП до картинки фона чтобы прога работала везде куда ее скачают

path3 = os.path.join(path[0:n] + "\Graphics", "red.gif") # ОП до картинки кнопки ред записи

path4 = os.path.join(path[0:n] + "\Graphics", "plus.gif") # ОП до картинки кнопки доб записи

path5 = os.path.join(path[0:n] + "\Graphics", "del.gif") # ОП до картинки кнопки удаление записи

path6 = os.path.join(path[0:n] + "\Graphics", "download.gif") # ОП до картинки кнопки загрузки бд

path7 = os.path.join(path[0:n] + "\Graphics", "save.gif") # ОП до картинки кнопки сохранения бд

path9 = os.path.join(path[0:n] + "\Library")

sys.path.append(path9)

from ourmodule import \*

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

def colors(path8, length):

"""

Функция считывания параметров интерфейса из файла

Автор: Животов Глеб

:param path8: Путь к текстовому файлу с параметрами

:param length: Количество цветов

:return: Список, состоящий из количества length цветов, названия шрифта и размера шрифта

"""

a = []

for i in range(length):

a.append("white")

f = open(path8, "r")

i = 0

for k in f:

k = k.rstrip('\n')

try:

a[i] = str(k)

except IndexError:

a.append(str(k))

i += 1

f.close()

return a

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

a = []

a = colors(path8, 4)

butcolor = a[0]

bgcolor1 = a[1]

bgcolor2 = a[2]

bgcolor3 = a[3]

mainfont = (str(a[4]), int(a[5]))

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

class MultiListbox(Frame):

def \_\_init\_\_(self, master, lists):

Frame.\_\_init\_\_(self, master)

self.lists = []

for l, w in lists:

frame = Frame(self)

frame.pack(side=LEFT, expand=YES, fill=BOTH)

Label(frame, text=l, borderwidth=1, relief=RAISED, bg=bgcolor2, font=mainfont).pack(fill=X)

lb = Listbox(frame, width=w, height = 30, borderwidth=0, selectborderwidth=0,

relief=FLAT, exportselection=FALSE, bg=bgcolor3, font=mainfont)

lb.pack(expand=YES, fill=BOTH)

self.lists.append(lb)

lb.bind('<B1-Motion>', lambda e, s=self: s.\_select(e.y))

lb.bind('<Button-1>', lambda e, s=self: s.\_select(e.y))

lb.bind('<Leave>', lambda e: 'break')

lb.bind('<B2-Motion>', lambda e, s=self: s.\_b2motion(e.x, e.y))

lb.bind('<Button-2>', lambda e, s=self: s.\_button2(e.x, e.y))

lb.bind('<Double-Button-1>', lambda e, s=self: print(self.get(self.curselection())[0]))

frame = Frame(self)

frame.pack(side=LEFT, fill=Y)

Label(frame, borderwidth=1, relief=RAISED).pack(fill=X)

sb = Scrollbar(frame, orient=VERTICAL, command=self.\_scroll)

sb.pack(expand=YES, fill=Y)

self.lists[0]['yscrollcommand'] = sb.set

def \_select(self, y):

row = self.lists[0].nearest(y)

self.selection\_clear(0, END)

self.selection\_set(row)

return 'break'

def \_button2(self, x, y):

for l in self.lists:

l.scan\_mark(x, y)

return 'break'

def \_b2motion(self, x, y):

for l in self.lists:

l.scan\_dragto(x, y)

return 'break'

def \_scroll(self, \*args):

for l in self.lists:

try:

l.yview(args[0], args[1], args[2])

except IndexError:

l.yview(args[0],args[1])

def curselection(self):

return self.lists[0].curselection()

def delete(self, first, last=None):

for l in self.lists:

l.delete(first, last)

def get(self, first, last=None):

if first == ():

return []

result = []

for l in self.lists:

result.append(l.get(first, last))

if last == first:

return map([None] + result)

return result

def index(self, index):

self.lists[0].index(index)

def insert(self, index, \*elements):

for e in elements:

i = 0

for l in self.lists:

l.insert(index, e[i])

i = i + 1

for l in self.lists:

l.see(END)

def size(self):

return self.lists[0].size()

def see(self, index):

for l in self.lists:

l.see(index)

def selection\_anchor(self, index):

for l in self.lists:

l.selection\_anchor(index)

def selection\_clear(self, first, last=None):

for l in self.lists:

l.selection\_clear(first, last)

def selection\_includes(self, index):

return self.lists[0].selection\_includes(index)

def selection\_set(self, first, last=None):

for l in self.lists:

l.selection\_set(first, last)

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

def load\_base\_button():

"""

Функция, вызываемая при нажатии кнопки загрузки. Вынужденное использование глобальной переменной из-за особенностей

вызова функции при нажатии кнопки. Загружает базу данных из двоичного файла в оперативную память.

Автор: Игуменова Марта

:return: None

"""

global w

w = byte(w, path1)

update\_output(multibox, w)

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

def open\_w\_summary(f, bgcolor1, bgcolor3, mainfont):

"""

Функция, вызываемая при нажатии кнопки подведения итого. Создаёт диалоговое с возможностью задания имени выходного

файла

Автор: Миронюк Даниил

:param f: Список - выборка записей, подходящих по параметру

:param bgcolor1: Цвет фона 1

:param bgcolor3: Цвет фона 2

:param mainfont: Список - Название шрифта и размер

:return: None

"""

askwindow = Tk()

askwindow.config(bg=bgcolor3)

askwindow.title("Сохранение файла")

asklabel = Label(askwindow, text = "Введите название файла", bg=bgcolor3, font=mainfont)

asklabel.grid(row=0)

askentry = Entry(askwindow)

askentry.grid(row=1)

askentry.insert(0, "saved")

askbutton = Button(askwindow, text="Сохранить файл", bg=bgcolor1, font=mainfont)

askbutton.bind("<Button-1>", lambda e: summary(f, askentry))

askbutton.grid(row=2)

askwindow.mainloop()

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

def open\_w\_findmensh(w, bgcolor1, bgcolor2, bgcolor3, mainfont):

"""

Функция, вызываемая при нажатии кнопки "Найти меньше". Создаёт окно для ввода интересующих данных и запуска функции

поиска записей со значением выбранного поля меньше заданного

Автор: Животов Глеб

:param w: Словарь словарей базы данных

:param bgcolor1: Цвет фона 1

:param bgcolor2: Цвет фона 2

:param bgcolor3: Цвет фона 3

:param mainfont: Список - Название шрифта, размер

:return: None

"""

bolshwind = Tk()

bolshwind.title("Найти записи меньше опредленого значения")

bolshwind.config(bg=bgcolor3)

message = Label(bolshwind, text="Выберете интересующий ключ и введите значение", bg=bgcolor3, font=mainfont)

message.grid(row=0, column=1)

kluchi = ["Год", "Масса", "Эф. масса", "Ступени"]

kluch = StringVar()

kluch.set(kluchi[0])

opt = OptionMenu(bolshwind, kluch, \*kluchi)

opt.config(bg=bgcolor2, font=mainfont)

opt.grid(row=1, column=0, sticky=EW)

edit = Entry(bolshwind, width=20)

edit.grid(row=1, column=1, sticky=EW)

edit.insert(0, "0")

def start(w, bgcolor1, bgcolor2, bgcolor3, mainfont):

"""

:param w:

:return:

"""

try:

num = int(edit.get())

except ValueError or TypeError:

mbox.showerror("Неверный ввод", "Введено неверное значение.")

return

message.grid\_remove()

opt.grid\_remove()

start\_function.grid\_remove()

edit.grid\_remove()

key = kluch.get()

if key == "Год":

key = "year"

elif key == "Масса":

key = "mass"

elif key == "Эф. масса":

key = "efmass"

elif key == "Ступени":

key = "stages"

f = find\_mensh(w, key, num)

label = Label(bolshwind, text="Записи с параметром " + str(key) + " меньше " + str(num), bg=bgcolor1, font=mainfont)

label.grid(row=0, column=0)

multibox = MultiListbox(bolshwind, (("Название", 17), ("Год", 17), ('Разработчик', 17), ('Масса', 17),

('Эфф. масса', 17), ('Ступени', 17)))

multibox.grid(row=1, column=0)

itogo = Button(bolshwind, text="Подвести итоги", bg=bgcolor2, font=mainfont)

itogo.grid(row=3, column=0, sticky=NSEW)

itogo.bind("<Button-1>", lambda e: open\_w\_summary(f, bgcolor1, bgcolor3, mainfont))

for i in f:

multibox.insert(END, (str(i['name']), str(i['year']), str(i['dev']), str(i['mass']),

str(i['efmass']), str(i['stages'])))

start\_function = Button(bolshwind, text="Старт", bg=bgcolor2, font=mainfont)

start\_function.grid(row=1, column=2, sticky=EW)

start\_function.bind("<Button-1>", lambda e: start(w, bgcolor1, bgcolor2, bgcolor3, mainfont))

bolshwind.mainloop()

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

def open\_w\_findbolsh(w, bgcolor1, bgcolor2, bgcolor3, mainfont):

"""

Функция, вызываемая при нажатии кнопки "Найти больше". Создаёт окно для ввода интересующих данных и запуска функции

поиска записей со значением выбранного поля больше заданного

Автор: Миронюк Даниил

:param w: Словарь словарей базы данных

:param bgcolor1: Цвет фона 1

:param bgcolor2: Цвет фона 2

:param bgcolor3: Цвет фона 3

:param mainfont: Список - Название фона, размер фона

:return: None

"""

bolshwind = Tk()

bolshwind.config(bg=bgcolor3)

bolshwind.title("Найти записи больше опредленого значения")

message = Label(bolshwind, text="Выберете интересующий ключ и введите значение", bg=bgcolor3, font=mainfont)

message.grid(row=0, column=1)

kluchi = ["Год", "Масса", "Эф. масса", "Ступени"]

kluch = StringVar()

kluch.set(kluchi[0])

opt = OptionMenu(bolshwind, kluch, \*kluchi)

opt.config(bg=bgcolor2, font=mainfont)

opt.grid(row=1, column=0, sticky=EW)

edit = Entry(bolshwind, width=20)

edit.grid(row=1, column=1, sticky=EW)

edit.insert(0, "0")

def start(w, bgcolor1, bgcolor2, bgcolor3, mainfont):

"""

:param w:

:return:

"""

try:

num = int(edit.get())

except ValueError or TypeError:

mbox.showerror("Неверный ввод", "Введено неверное значение.")

return

message.grid\_remove()

opt.grid\_remove()

start\_function.grid\_remove()

edit.grid\_remove()

key = kluch.get()

if key == "Год":

key = "year"

elif key == "Масса":

key = "mass"

elif key == "Эф. масса":

key = "efmass"

elif key == "Ступени":

key = "stages"

f = find\_bolsh(w, key, num)

label = Label(bolshwind, text="Записи с параметром " + str(key) + " больше " + str(num), bg=bgcolor1, font=mainfont)

label.grid(row=0, column=0)

multibox = MultiListbox(bolshwind, (("Название", 17), ("Год", 17), ('Разработчик', 17), ('Масса', 17),

('Эфф. масса', 17), ('Ступени', 17)))

multibox.grid(row=1, column=0)

itogo = Button(bolshwind, text="Подвести итоги", bg=bgcolor2, font=mainfont)

itogo.grid(row=3, column=0, sticky=NSEW)

itogo.bind("<Button-1>", lambda e: open\_w\_summary(f, bgcolor1, bgcolor3, mainfont))

for i in f:

multibox.insert(END, (str(i['name']), str(i['year']), str(i['dev']), str(i['mass']),

str(i['efmass']), str(i['stages'])))

start\_function = Button(bolshwind, text="Старт", bg = bgcolor2, font=mainfont)

start\_function.grid(row=1, column=2, sticky=EW)

start\_function.bind("<Button-1>", lambda e: start(w, bgcolor1, bgcolor2, bgcolor3, mainfont))

bolshwind.mainloop()

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

def open\_w\_findbetween(w, bgcolor1, bgcolor2, bgcolor3, mainfont):

"""

Функция, вызываемая при нажатии кнопки "Найти в диапазоне". Создаёт окно для ввода интересующих данных и запуска функции

поиска записей со значением выбранного поля в заданном диапазоне

Автор: Игуменова Марта

:param w: Словарь словарей базы данных

:param bgcolor1: Цвет фона 1

:param bgcolor2: Цвет фона 2

:param bgcolor3: Цвет фона 3

:param mainfont: Список - Название шрифта, размер шрифта

:return: None

"""

betweenwind = Tk()

betweenwind.config(bg=bgcolor3)

betweenwind.title("Найти записи меньше опредленого значения")

message = Label(betweenwind, text="Выберете интересующий ключ и диапазон", bg=bgcolor3, font=mainfont)

message.grid(row=0, column=0, columnspan=3)

ot = Label(betweenwind, text="От", bg=bgcolor3, font=mainfont)

ot.grid(row=1, column=0)

edit1 = Entry(betweenwind)

edit1.grid(row=1, column=1)

edit1.insert(0, '0')

do = Label(betweenwind, text="До", bg=bgcolor3, font=mainfont)

do.grid(row=2, column=0)

edit2 = Entry(betweenwind)

edit2.grid(row=2, column=1)

edit2.insert(0, "0")

kluchi = ["Год", "Масса", "Эф. масса", "Ступени"]

kluch = StringVar()

kluch.set(kluchi[0])

opt = OptionMenu(betweenwind, kluch, \*kluchi)

opt.config(bg=bgcolor2)

opt.grid(row=3, column=0, columnspan=3, sticky=NSEW)

def start(w, bgcolor1, bgcolor2, bgcolor3, mainfont):

"""

:param w:

:return:

"""

try:

num1 = int(edit1.get())

num2 = int(edit2.get())

except ValueError or TypeError:

mbox.showerror("Неверный ввод", "Введено неверное значение.")

return

start\_function.grid\_remove()

opt.grid\_remove()

edit1.grid\_remove()

edit2.grid\_remove()

ot.grid\_remove()

do.grid\_remove()

message.grid\_remove()

key = kluch.get()

if key == "Год":

key = "year"

elif key == "Масса":

key = "mass"

elif key == "Эф. масса":

key = "efmass"

elif key == "Ступени":

key = "stages"

f = find\_between(w, key, num1, num2)

label = Label(betweenwind, text="Записи с параметром " + str(key) + " от " + str(num1) + " до " + str(num2), bg=bgcolor3, font=mainfont)

label.grid(row=0)

itogo = Button(betweenwind, text="Подвести итоги", bg=bgcolor2, font=mainfont)

itogo.grid(row=14, column=0, sticky=NSEW)

itogo.bind("<Button-1>", lambda e: open\_w\_summary(f, bgcolor1, bgcolor3, mainfont))

multibox = MultiListbox(betweenwind, (("Название", 17), ("Год", 17), ('Разработчик', 17), ('Масса', 17),

('Эфф. масса', 17), ('Ступени', 17)))

multibox.grid(row=2)

for i in f:

multibox.insert(END, (str(i['name']), str(i['year']), str(i['dev']), str(i['mass']),

str(i['efmass']), str(i['stages'])))

start\_function = Button(betweenwind, text="Старт", bg=bgcolor1, font=mainfont)

start\_function.bind("<Button-1>", lambda e: start(w, bgcolor1, bgcolor2, bgcolor3, mainfont))

start\_function.grid(row=4, column=0, sticky=NSEW, columnspan=3)

betweenwind.mainloop()

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

def open\_w\_findname(w, bgcolor1, bgcolor2, bgcolor3, mainfont):

"""

Функция, вызываемая при нажатии кнопки "Поиск по имени". Создаёт окно для ввода интересующих данных и запуска функции

поиска записей с заданным именем

Автор: Животов Глеб

:param w: Словарь словарей базы данных

:param bgcolor1: Цвет фона 1

:param bgcolor2: Цвет фона 2

:param bgcolor3: Цвет фона 3

:param mainfont: Список - Название шрифта, размер шрифта

:return: None

"""

findname= Tk()

findname.config(bg=bgcolor3)

findname.title("Найти записи c именем")

message = Label(findname, text="Введите имя", bg=bgcolor3, font=mainfont)

message.grid(row=0, column=0, sticky=NSEW)

edit = Entry(findname)

edit.grid(row=1, column=0, sticky=NSEW)

def start(w, bgcolor1, bgcolor2, bgcolor3, mainfont):

name = edit.get()

if name == '':

mbox.showerror("Пустое имя", "Введено пустое имя")

return

start\_function.grid\_remove()

edit.grid\_remove()

message.grid\_remove()

f = find\_name(w, name)

label = Label(findname, text="Записи с именем " + str(name), bg=bgcolor3, font=mainfont)

label.grid(row=0)

itogo = Button(findname, text="Подвести итоги", bg=bgcolor2, font=mainfont)

itogo.grid(row=14, column=0, sticky=NSEW)

itogo.bind("<Button-1>", lambda e: open\_w\_summary(f, bgcolor1, bgcolor3, mainfont))

multibox = MultiListbox(findname, (("Название", 17), ("Год", 17), ('Разработчик', 17), ('Масса', 17),

('Эфф. масса', 17), ('Ступени', 17)))

multibox.grid(row=2)

for i in f:

multibox.insert(END, (str(i['name']), str(i['year']), str(i['dev']), str(i['mass']),

str(i['efmass']), str(i['stages'])))

start\_function = Button(findname, text="Старт", bg=bgcolor1, font=mainfont)

start\_function.grid(row=3, column=0, sticky=NSEW)

start\_function.bind("<Button-1>", lambda e: start(w, bgcolor1, bgcolor2, bgcolor3, mainfont))

findname.mainloop()

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

def click\_add\_node(w, namee, yeare, deve, masse, efmasse, stagese, multibox):

"""

Функция, вызываемая при нажатии кнопки добавления новой записи. Считывает данные из полей и запускает функцию

добавления записи

Автор: Миронюк Даниил

:param w: Словарь словарей базы данных

:param namee: Объект класса Entry, содержащий новое имя

:param yeare: Объект класса Entry, содержащий новый год

:param deve: Объект класса Entry, содержащий нового разработчик

:param masse: Объект класса Entry, содержащий новую массу

:param efmasse: Объект класса Entry, содержащий новую эффективную массу

:param stagese: Объект класса Entry, содержащий новое количество степеней

:param multibox: Объект класса Multilistbox, который будет обновлён после добавления записи

:return: None

"""

name = namee.get()

year = yeare.get()

dev = deve.get()

mass = masse.get()

efmass = efmasse.get()

stages = stagese.get()

if name == '' or year == '' or dev == '' or mass == '' or efmass == '' or stages == '':

mbox.showerror("Пустые поля", "Заполните поля перед добавлением записи")

return

add\_node(w, name, year, dev, mass, efmass, stages)

update\_output(multibox, w)

namee.delete(0, END)

yeare.delete(0, END)

deve.delete(0, END)

masse.delete(0, END)

efmasse.delete(0, END)

stagese.delete(0, END)

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

def click\_change\_node(w, namee, yeare, deve, masse, efmasse, stagese, multibox, key):

"""

Функция, вызываемая при нажатии кнопки изменения записи. Считывает данные полей и запускает функцию изменения записи

Автор: Животов Глеб

:param w: Словарь словарей базы данных

:param namee: Объект класса Entry, содержащий новое имя

:param yeare: Объект класса Entry, содержащий новый год

:param deve: Объект класса Entry, содержащий нового разработчик

:param masse: Объект класса Entry, содержащий новую массу

:param efmasse: Объект класса Entry, содержащий новую эффективную массу

:param stagese: Объект класса Entry, содержащий новое количество степеней

:param multibox: Объект класса Multilistbox, который будет обновлён после добавления записи

:param key: Список полей записи, где 0 элемент - искомый ключ

:return: None

"""

try:

key = key[0]

except IndexError:

mbox.showerror("Отсутсвует выделение", "Не выбрана строка для редактирования. Выберите и попробуйте снова")

return

name = namee.get()

year = yeare.get()

dev = deve.get()

mass = masse.get()

efmass = efmasse.get()

stages = stagese.get()

if name == '' or year == '' or dev == '' or mass == '' or efmass == '' or stages == '':

mbox.showerror("Пустые поля", "Заполните поля перед редактированием записи")

return

change\_node(w, key, name, year, dev, mass, efmass, stages)

update\_output(multibox, w)

namee.delete(0, END)

yeare.delete(0, END)

deve.delete(0, END)

masse.delete(0, END)

efmasse.delete(0, END)

stagese.delete(0,END)

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

def open\_w\_save(w, path1):

"""

Автор: Игуменова Марта

Функция сохранения словаря словарей в двоичный файл

:param w: Словарь словарей базы данных

:param path1: Путь к двоичному файлу

:return: None

"""

ans = mbox.askokcancel("Сохранить базу данных?", "Двоичный файл будет изменён. Вы уверены?")

if ans:

f = open(path1, "wb")

pickle.dump(w, f)

f.close()

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

w ={}

root = Tk()

root.title("STAR BASE BIZARU ADVENTURES VER. 08.06")

img\_change = PhotoImage(file=path3)

img\_plus = PhotoImage(file=path4)

img\_del = PhotoImage(file=path5)

img\_load = PhotoImage(file=path6)

img\_save = PhotoImage(file=path7)

filename = PhotoImage(file=path2)

background\_label = Label(root, image=filename)

background\_label.place(x=0, y=0, relwidth=1, relheight=1)

buttonframe = Frame(root)

buttonframe.grid(row=0, column=1, columnspan=8)

multiframe = Frame(root)

multiframe.grid(row=1, column=1, columnspan=8)

editframe = Frame(root, bg=bgcolor1)

editframe.grid(row=0, column=10, rowspan=33)

picbuttonframe = Frame(root)

picbuttonframe.grid(row=0, column=0, rowspan=33)

multibox = MultiListbox(multiframe, (("ID", 5), ("Название", 17), ("Год", 17), ('Разработчик', 17), ('Масса', 17),

('Эфф. масса', 17), ('Ступени', 17)))

multibox.pack()

MenshButton = Button(buttonframe, text="Найти меньше", width=15, bg=butcolor, font=mainfont)

MenshButton.pack(side=LEFT)

MenshButton.bind("<Button-1>", lambda e: open\_w\_findmensh(w, bgcolor1, bgcolor2, bgcolor3, mainfont))

BolshButton = Button(buttonframe, text="Найти больше ", width=15, bg=butcolor, font=mainfont)

BolshButton.pack(side=LEFT)

BolshButton.bind("<Button-1>", lambda e: open\_w\_findbolsh(w, bgcolor1, bgcolor2, bgcolor3, mainfont))

BetwButton = Button(buttonframe, text="Найти в диапазоне", width=15, bg=butcolor, font=mainfont)

BetwButton.pack(side=LEFT)

BetwButton.bind("<Button-1>", lambda e: open\_w\_findbetween(w, bgcolor1, bgcolor2, bgcolor3, mainfont))

NameButton = Button(buttonframe, text="Поиск по имени", width=15, bg=butcolor, font=mainfont)

NameButton.pack(side=LEFT)

NameButton.bind("<Button-1>", lambda e: open\_w\_findname(w, bgcolor1, bgcolor2, bgcolor3, mainfont))

namel = Label(editframe, text="Название", bg=bgcolor1, font=mainfont)

namel.grid(row=0, column=0)

namee = Entry(editframe, bg=bgcolor1)

namee.grid(row=1, column=0)

yearl = Label(editframe, text="Год", bg=bgcolor1, font=mainfont)

yearl.grid(row=2, column=0)

yeare = Entry(editframe, bg=bgcolor1)

yeare.grid(row=3, column=0)

devl = Label(editframe, text="Разработчик", bg=bgcolor1, font=mainfont)

devl.grid(row=4, column=0)

deve = Entry(editframe, bg=bgcolor1)

deve.grid(row=5, column=0)

massl = Label(editframe, text="Масса", bg=bgcolor1, font=mainfont)

massl.grid(row=6, column=0)

masse = Entry(editframe, bg=bgcolor1)

masse.grid(row=7, column=0)

efmassl = Label(editframe, text="Эффективная масса", bg=bgcolor1, font=mainfont)

efmassl.grid(row=8, column=0)

efmasse = Entry(editframe, bg=bgcolor1)

efmasse.grid(row=9, column=0)

stagesl = Label(editframe, text="Количество ступеней", bg=bgcolor1, font=mainfont)

stagesl.grid(row=10, column=0)

stagese = Entry(editframe, bg=bgcolor1)

stagese.grid(row=11, column=0)

LoadBaseBut = Button(picbuttonframe, image=img\_load)

LoadBaseBut.grid(row=5, column=0)

LoadBaseBut.bind("<Button-1>", lambda e: load\_base\_button())

ChangeButton=Button(picbuttonframe, image=img\_change)

ChangeButton.grid(row=6, column=0)

ChangeButton.bind("<Button-1>", lambda e: click\_change\_node(w, namee, yeare, deve, masse, efmasse, stagese, multibox,

multibox.get(multibox.curselection())))

AddButton = Button(picbuttonframe, image=img\_plus)

AddButton.grid(row=7, column=0)

AddButton.bind("<Button-1>", lambda e: click\_add\_node(w, namee, yeare, deve, masse, efmasse, stagese, multibox))

DelButton = Button(picbuttonframe, image=img\_del)

DelButton.grid(row=8, column=0)

DelButton.bind("<Button-1>", lambda e: delete\_node(w, multibox.get(multibox.curselection()), multibox))

SaveButton = Button(picbuttonframe, image=img\_save)

SaveButton.grid(row=9, column=0)

SaveButton.bind("<Button-1>", lambda e: open\_w\_save(w, path1))

update\_output(multibox, w)

root.mainloop()

Приложение 3

Содержание файла «colors.txt»

#0c04b9

#534aff

#3228ff

#c7c4ff

Arial

10