

1. Полиморфизм в языках программирования.

Полиморфизм в языках программирования — возможность объектов с одинаковой спецификацией иметь различную реализацию. Происходит от греческих слов “Poly” – множество “Morphism” – форм. Делает процесс программирования легче и интуитивнее.

Пример полиморфизма с наследованием показан в коде программы Polymorphism1.py. Определяет методы в классе – приемнике, имеющие такое же имя, как и методы в классе – родителе. Такие методы по наследству переходят к классу приемнику от класса – родителя. Также имеется возможность изменять методы в классе – приемнике, унаследованные от класса – родителя. В основном используется в случаях, где методы, унаследованные от класса – родителя не подходят классу – приемнику. Такой процесс реимплементации метода в классе – приемнике имеет название «Перезапись Метода».

Код программы Polymorphism1.py

class Bird:

def intro(self):

print("Существуют разные виды птиц")

def flight(self):

print("Большинство птиц умеют летать, но некоторые - нет")

class parrot(Bird):

def flight(self):

print("Попугаи умеют летать")

class penguin(Bird):

def flight(self):

print("Пингвины не умеют летать")

obj\_bird = Bird()

obj\_parr = parrot()

obj\_peng = penguin()

obj\_bird.intro()

obj\_bird.flight()

#пример полиформизма с наследством

#происходит обращение к методам класса - родителя

obj\_parr.intro()

obj\_parr.flight()

obj\_peng.intro()

obj\_peng.flight()

Пример полиморфизма времени компиляции через перегрузку функций представлен в коде программы Polymorphism2.cpp. Перегрузка функций происходит, когда существует несколько функций с одинаковыми именами, при обращении к одной из них через особое количество, или специальный тип переменных.

Код программы Polymorphism1.py

/// программа на c++ по перезаписи функций

#include <iostream>

using namespace std;

class PolymorphFO

{

public:

/// функция с одним параметром типа int

void func(int x)

{

cout << "Величина x: " << x << endl;

}

/// функция с таким же названием, но параметром типа double

void func(double x)

{

cout << "Величина x: " << x << endl;

}

/// функция с таким же именем, но двумя параметрами типа int

void func(int x, int y)

{

cout << "Величина x и y: " << x << ", " << y << endl;

}

};

int main() {

setlocale(LC\_ALL, "Russian");

PolymorphFO obj1;

/// то, какая функция будет вызвана, определяет тип передаваемых данных

/// первая 'func' вызвана

obj1.func(7);

/// вызвана вторая 'func'

obj1.func(9.132);

/// вызвана третья 'func'

obj1.func(85,64);

return 0;

}

1. Текстовый файл содержит слова, разделенные одним или несколькими пробелами. Разработать приложение для построения отсортированного словаря используемых слов. Вывести словарь в файл, используя SaveFileDialog.

К файлу word прилагаются так же файл с программой (Exam\_Ticket22\_Ovchiinikov.py) и необходимый для работы текстовый файл (unsorted.txt).

Код разработанного приложения:

# импорт всех файлов с библиотеки tkinter

from tkinter import \*

from tkinter import ttk

# импорт только asksaveasfile из filedialog

# позволяет сохранить файл в любом расширении

from tkinter.filedialog import asksaveasfile

root = Tk()

root.geometry('200x250')

""" функция вызывается, когда пользователь нажимает на

кнопку 'сохранить'. Появляется окно для выбора места сохранения"""

def save():

files = [('All Files', '\*.\*'),

('Text Document', '\*.txt')]

file = asksaveasfile(filetypes=files, defaultextension=files)

for i in range(len(list)):

file.write(list[i])

file.write('\n')

file.close()

f = open('unsorted.txt', 'r')

list = []

# считываем каждую линию

for line in f:

# считываем каждое слово

for word in line.split():

list.append(word)

# сортируем список всех слов

list.sort()

# создаём виджет для отображения сортированного текста

listbox = Listbox(root)

listbox.pack()

# добавляем в окно для текста сортированный текст

for item in list:

listbox.insert(END, item)

# создаём ползунок для полного отображения текста в виджете с текстом

scroll = Scrollbar(command=listbox.yview)

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

listbox.config(yscrollcommand=scroll.set)

btn = ttk.Button(root, text='Сохранить', command=lambda: save())

btn.pack(side=TOP, pady=20)

mainloop()