

Практическое занятие №17

**Тема:** составление программ с использованием GUI Tkinter в IDE PyCharm Community, изучение возможностей модуля OS.

**Цели:** закрепить усвоенные знания, понятия, алгоритмы, основные принципы составления программ, приобрести навыки составления программ с использованием GUI Tkinter в IDE PyCharm Community, изучить возможности модуля OS.

**Постановка задачи:**

1. В соответствии с номером варианта перейти по ссылке на прототип. Реализовать его в IDE PyCharm Community с применением пакета tk. Получить интерфейс максимально приближенный к оригиналу (см. таблицу 1).
2. Разработать программу с применением пакета tk, взяв в качестве условия одну любую задачу из ПЗ № 2 – 9.
3. Задание предполагает, что у студента есть проект с практическими работами (№ 2-13), оформленный согласно требованиям. Все задания выполняются с использованием модуля OS:
  - перейдите в каталог PZ11. Выведите список всех файлов в этом каталоге. Имена вложенных подкаталогов выводить не нужно.
  - перейти в корень проекта, создать папку с именем test. В ней создать еще одну папку test1. В папку test переместить два файла из ПЗ6, а в папку test1 - один файл из ПЗ7. Файл из ПЗ7 переименовать в test.txt. Вывести в консоль информацию о размере файлов в папке test.
  - перейти в папку с PZ11, найти там файл с самым коротким именем, имя вывести в консоль. Использовать функцию `basename()` (`os.path.basename()`).
  - перейти в любую папку где есть отчет в формате .pdf и «запустите» файл в привязанной к нему программе. Использовать функцию `startfile()`.
  - удалить файл test.txt

**Текст программы:**

1.

```

1  ✓ #В соответствии с номером варианта перейти по ссылке на прототип. Реализовать
2  #его в IDE PyCharm Community с применением пакета tk. Получить интерфейс максимал
3  #приближенный к оригиналу (см. таблицу 1).
4  ✓ import tkinter as tk
5  from tkinter import ttk
6  from datetime import datetime
7  from pprint import pprint
8
9  1 usage  👤 victoriaskobelina
10  ✓ def window_deleted():
11      print("Окно закрыто")
12      root.quit()
13
14  2 usages  👤 victoriaskobelina *
15  ✓ def cancel():
16      entry_first_name.delete( first: 0, tk.END)
17      entry_first_name.config(fg='Black')
18      focus_out_entry_box(entry_first_name, entry_first_name_text)
19      entry_last_name.delete( first: 0, tk.END)
20      entry_last_name.config(fg='Black')
21      focus_out_entry_box(entry_last_name, entry_last_name_text)
22      entry_screen_name.delete( first: 0, tk.END)
23      entry_screen_name.config(fg='Black')
24
25      focus_out_entry_box(entry_screen_name, entry_screen_name_text)
26      entry_email.delete( first: 0, tk.END)
27      entry_email.config(fg='Black')
28      focus_out_entry_box(entry_email, entry_email_text)
29      entry_phone.delete( first: 0, tk.END)
30      entry_phone.config(fg='Black')
31      focus_out_entry_box(entry_phone, entry_phone_text)
32      entry_password.delete( first: 0, tk.END)
33      entry_confirm_password.delete( first: 0, tk.END)
34      default_month.set(month_list[0])
35      default_day.set(day_list[0])
36      default_year.set("1985")
37      default_country.set(country_list[0])
38      radio_button_male.config(variable=tk.StringVar())
39      radio_button_female.config(variable=tk.StringVar())
40      checkbox_terms_of_use.config(variable=tk.BooleanVar(value=False))
41      print("Форма очищена\n")
42
43  1 usage  👤 victoriaskobelina
44  def submit():
45      gotten_month_num = str(month_list.index(combo_box_month.get()) + 1).rjust( __width: 2, __fill:

```

```

42     gotten_year = combo_box_year.get()
43     gotten_day = combo_box_day.get().rjust( _width: 2, _fillchar: "0")
44     try:
45         full_date = datetime.strptime( _date_string: f"{gotten_year}-{gotten_month_num}-{gotten_day}", _format: "%
46     except ValueError:
47         full_date = ""
48         print("Invalid date was given!\n")
49
50     data = {
51         "first_name": entry_first_name.get(),
52         "last_name": entry_last_name.get(),
53         "screen_name": entry_screen_name.get(),
54         "email": entry_email.get(),
55         "phone": entry_phone.get(),
56         "password": entry_password.get(),
57         "confirm_password": entry_confirm_password.get(),
58         "date_of_birth": full_date,
59         "country": combo_box_country.get(),
60         "gender": gender.get(),
61         "terms_agree": terms_agree.get(),
62     }
63
64     pprint(data, indent=4)
65     print("\n")
66     cancel()
67
68     #ОСНОВНЫЕ ЭЛЕМЕНТЫ
69     root = tk.Tk()
70     root.title("Sign Up")
71     root.geometry("560x660+700+400")
72     root.protocol( name: 'WM_DELETE_WINDOW', window_deleted) #обработчик закрытия окна
73     root.resizable( width: False, height: False) #размер окна не изменяется
74     root.config(bg="#de8704")
75     for i in range(10):
76         root.columnconfigure(index=i, weight=1)
77     for i in range(23):
78         root.rowconfigure(index=i, weight=1)
79     main_title_label = tk.Label(root, text='Sign Up', bg="#de8704", fg="#fde82d", font=("Arial", 16))
80     main_frame = tk.Frame(root, bg="#222536", bd=5)
81     cancel_button = tk.Button(root, text='Cancel', font=("Arial", 11, "bold"), background="red", foreground="white", command=
82     submit_button = tk.Button(root, text='Submit', font=("Arial", 11, "bold"), background="green", foreground="white", comman
83     main_title_label.place(x=10, y=4)
84     main_frame.grid(row=1, column=0, columnspan=11, rowspan=22, padx=0, pady=12, sticky="nsew")
85
86     cancel_button.grid(row=23, column=9, columnspan=1, ipadx=4, ipady=4, padx=0, pady=12)
87     submit_button.grid(row=23, column=10, columnspan=1, ipadx=4, ipady=4, padx=12, pady=12)
88
89     #ЯРЛЫКИ ДЛЯ ПОЛЕЙ
90     first_name_label = tk.Label(root, text='First Name', bg="#222536", fg="#fde82d", font=("Arial", 11, "bold"))
91     first_name_label.grid(row=1, column=0, columnspan=4, rowspan=2, padx=0, pady=0, sticky="e")
92     last_name_label = tk.Label(root, text='Last Name', bg="#222536", fg="#fde82d", font=("Arial", 11, "bold"))
93     last_name_label.grid(row=3, column=0, columnspan=4, rowspan=2, padx=0, pady=0, sticky="e")
94     screen_name_label = tk.Label(root, text='Screen Name', bg="#222536", fg="#fde82d", font=("Arial", 11, "bold"))
95     screen_name_label.grid(row=5, column=0, columnspan=4, rowspan=2, padx=0, pady=0, sticky="e")
96     date_of_birth_label = tk.Label(root, text='Date Of Birth', bg="#222536", fg="#fde82d", font=("Arial", 11, "bold"))
97     date_of_birth_label.grid(row=7, column=0, columnspan=4, rowspan=2, padx=0, pady=0, sticky="e")
98     gender_label = tk.Label(root, text='Gender', bg="#222536", fg="#fde82d", font=("Arial", 11, "bold"))
99     gender_label.grid(row=9, column=0, columnspan=4, rowspan=2, padx=0, pady=0, sticky="e")
100     country_label = tk.Label(root, text='Country', bg="#222536", fg="#fde82d", font=("Arial", 11, "bold"))
101     country_label.grid(row=11, column=0, columnspan=4, rowspan=2, padx=0, pady=0, sticky="e")
102     email_label = tk.Label(root, text='E-mail', bg="#222536", fg="#fde82d", font=("Arial", 11, "bold"))
103     email_label.grid(row=13, column=0, columnspan=4, rowspan=2, padx=0, pady=0, sticky="e")
104     phone_label = tk.Label(root, text='Phone', bg="#222536", fg="#fde82d", font=("Arial", 11, "bold"))
105     phone_label.grid(row=15, column=0, columnspan=4, rowspan=2, padx=0, pady=0, sticky="e")
106     password_label = tk.Label(root, text='Password', bg="#222536", fg="#fde82d", font=("Arial", 11, "bold"))

```

## Студентка группы ИС-25 Скобелина В.В.

```
105 password_label.grid(row=17, column=0, columnspan=4, rowspan=2, padx=0, pady=0, sticky="e")
106 confirm_password_label = tk.Label(root, text='Confirm Password', bg="#222536", fg="#fde82d", font=("Arial", 11, "b"))
107 confirm_password_label.grid(row=19, column=0, columnspan=4, rowspan=2, padx=0, pady=0, sticky="e")
108
109 #ПОЛЯ ДЛЯ ЗАПОЛНЕНИЯ
110 10 usages victoriaskobelina
110 def focus_out_entry_box(widget, widget_text):
111     if widget['fg'] == 'Black' and len(widget.get()) == 0:
112         widget.delete(0, tk.END)
113         widget['fg'] = 'Grey'
114         widget.insert(0, widget_text)
115 5 usages victoriaskobelina
115 def focus_in_entry_box(widget):
116     if widget['fg'] == 'Grey':
117         widget['fg'] = 'Black'
118         widget.delete(0, tk.END)
119
120 entry_first_name_text = 'Enter First Name...'
121 entry_first_name = tk.Entry(root, font='Arial 11', fg='Grey')
122 entry_first_name.insert(index=0, entry_first_name_text)
123 entry_first_name.grid(row=1, column=5, columnspan=6, rowspan=2, padx=6, pady=0, sticky="we")
124
125 entry_first_name.bind("<FocusIn>", lambda args: focus_in_entry_box(entry_first_name))
126 entry_first_name.bind("<FocusOut>", lambda args: focus_out_entry_box(entry_first_name, entry_first_name_text))
127
128 entry_last_name_text = 'Enter Last Name...'
129 entry_last_name = tk.Entry(root, font='Arial 11', fg='Grey')
130 entry_last_name.insert(index=0, entry_last_name_text)
131 entry_last_name.grid(row=3, column=5, columnspan=6, rowspan=2, padx=6, pady=0, sticky="we")
132 entry_last_name.bind("<FocusIn>", lambda args: focus_in_entry_box(entry_last_name))
133 entry_last_name.bind("<FocusOut>", lambda args: focus_out_entry_box(entry_last_name, entry_last_name_text))
134
135 entry_screen_name_text = 'Enter Screen Name...'
136 entry_screen_name = tk.Entry(root, font='Arial 11', fg='Grey')
137 entry_screen_name.insert(index=0, entry_screen_name_text)
138 entry_screen_name.grid(row=5, column=5, columnspan=6, rowspan=2, padx=6, pady=0, sticky="we")
139 entry_screen_name.bind("<FocusIn>", lambda args: focus_in_entry_box(entry_screen_name))
140 entry_screen_name.bind("<FocusOut>", lambda args: focus_out_entry_box(entry_screen_name, entry_screen_name_text))
141
142 entry_email_text = 'Enter E-mail...'
143 entry_email = tk.Entry(root, font='Arial 11', fg='Grey')
144 entry_email.insert(index=0, entry_email_text)
145 entry_email.grid(row=13, column=5, columnspan=6, rowspan=2, padx=6, pady=0, sticky="we")
146
147 entry_email.bind("<FocusIn>", lambda args: focus_in_entry_box(entry_email))
148 entry_email.bind("<FocusOut>", lambda args: focus_out_entry_box(entry_email, entry_email_text))
149
150 entry_phone_text = 'Enter Phone...'
151 entry_phone = tk.Entry(root, font='Arial 11', fg='Grey')
152 entry_phone.insert(index=0, entry_phone_text)
153 entry_phone.grid(row=15, column=5, columnspan=6, rowspan=2, padx=6, pady=0, sticky="we")
154 entry_phone.bind("<FocusIn>", lambda args: focus_in_entry_box(entry_phone))
155 entry_phone.bind("<FocusOut>", lambda args: focus_out_entry_box(entry_phone, entry_phone_text))
156
157 entry_password = tk.Entry(root, font='Arial 11')
158 entry_password.grid(row=17, column=5, columnspan=6, rowspan=2, padx=6, pady=0, sticky="we")
159
160 entry_confirm_password = tk.Entry(root, font='Arial 11')
161 entry_confirm_password.grid(row=19, column=5, columnspan=6, rowspan=2, padx=6, pady=0, sticky="we")
162
163 #ПОЛЯ С ВЫБОРОМ
164 month_list = ["January", "February", "March", "April", "May", "June", "July", "August", "September", "October", "November", "December"]
165 default_month = tk.StringVar(value=month_list[0])
166 day_list = [str(i) for i in range(1, 32)]
```

```

166 default_day = tk.StringVar(value=day_list[0])
167 year_list = [str(i) for i in range(1900, 2025)]
168 default_year = tk.StringVar(value="1985")
169 country_list = ["USA", "Russian Federation", "Canada", "China", "Brazil", "Australia", "India", "Other"]
170 default_country = tk.StringVar(value=country_list[0])
171 combo_box_month = ttk.Combobox(root, values=month_list, textvariable=default_month, font='Arial 10 bold')
172 combo_box_month.grid(row=7, column=5, columnspan=4, rowspan=2, padx=6, pady=0, sticky="we")
173 combo_box_day = ttk.Combobox(root, values=day_list, textvariable=default_day, font='Arial 11', width=5)
174 combo_box_day.grid(row=7, column=9, columnspan=1, rowspan=2, padx=6, pady=0, sticky="we")
175 combo_box_year = ttk.Combobox(root, values=year_list, textvariable=default_year, font='Arial 11', width=5)
176 combo_box_year.grid(row=7, column=10, columnspan=2, rowspan=2, padx=6, pady=0, sticky="we")
177 combo_box_country = ttk.Combobox(root, values=country_list, textvariable=default_country, font='Arial 10 bold')
178 combo_box_country.grid(row=11, column=5, columnspan=6, rowspan=2, padx=6, pady=0, sticky="we")
179
180 #КНОПКИ ДЛЯ ВЫБОРА
181 male = "Male"
182 female = "Female"
183 gender = tk.StringVar()
184 gender_style = ttk.Style()
185 gender_style.configure(style="BW.TRadiobutton", font=("Arial", 9), background="#222536", border=0, foreground="#fde82d")
186 radio_button_male = ttk.Radiobutton(root, text="Male", value="male", variable=gender, style="BW.TRadiobutton")
187 radio_button_male.grid(row=9, column=5, columnspan=3, rowspan=2, padx=6, pady=0, sticky="w")
188 radio_button_female = ttk.Radiobutton(root, text="Female", value="female", variable=gender, style="BW.TRadiobutton")
189 radio_button_female.grid(row=9, column=8, columnspan=3, rowspan=2, padx=6, pady=0, sticky="w")
190 terms_agree = tk.BooleanVar(value=False)
191 terms_style = ttk.Style()
192 terms_style.configure(style="BW.TCheckbutton", font=("Arial", 9), background="#222536", border=0, foreground="#fde82d")
193 checkbutton_terms_of_use = ttk.Checkbutton(root, text="I agree to the Terms of Use", variable=terms_agree, style="BW.TCheckbutton")
194 checkbutton_terms_of_use.grid(row=21, column=5, columnspan=6, rowspan=1, padx=6, pady=0, sticky="w")
195
196 root.mainloop()

```

2.

```

2 #Разработать программу с применением пакета tk, взяв в качестве условия одну
  #любую задачу из ПЗ № 2 - 9.
3 #Известно, что x кг шоколадных конфет стоит a рублей, а y кг ирисок стоит b рублей.
4 #Определить сколько стоит 1 кг шоколадных конфет, 1 кг ирисок, а также во сколько раз шоколадные конфеты дороже ирисок.
5 import tkinter as tk
  usage = victoriaskobelina
6 def calculate_price():
7     x = float(entry_x.get())
8     a = float(entry_a.get())
9     y = float(entry_y.get())
10    b = float(entry_b.get())
11
12    price_choco_kg = a / x
13    price_iris_per_kg = b / y
14    price_ratio = price_choco_kg / price_iris_per_kg
15
16    result_label.config(text=f"Цена за 1 кг шоколадных конфет: {price_choco_kg:.2f} рублей\n"
17                          f"Цена за 1 кг ирисок: {price_iris_per_kg:.2f} рублей\n"
18                          f"Шоколадные конфеты дороже ирисок в {price_ratio:.2f} раз(a)")
19
20    root = tk.Tk()
21    root.title("Расчет цен на шоколадные конфеты и ириски")
22    tk.Label(root, text="Введите, пожалуйста, сколько x-кг шоколадных конфет:").pack()

```

```
23 entry_x = tk.Entry(root)
24 entry_x.pack()
25 tk.Label(root, text="Введите, пожалуйста, цену а за них:").pack()
26 entry_a = tk.Entry(root)
27 entry_a.pack()
28 tk.Label(root, text="Введите, пожалуйста, сколько у-кг ирисок:").pack()
29 entry_y = tk.Entry(root)
30 entry_y.pack()
31 tk.Label(root, text="Введите, пожалуйста, цену б за них:").pack()
32 entry_b = tk.Entry(root)
33 entry_b.pack()
34 calculate_button = tk.Button(root, text="Рассчитать", command=calculate_price)
35 calculate_button.pack()
36 result_label = tk.Label(root, text="")
37 result_label.pack()
38
39 root.mainloop()
```

**Протокол работы программы:**

**Sign Up**

**First Name**

**Last Name**

**Screen Name**

**Date Of Birth**

**Gender** ☐ Male ☐ Female

**Country**

**E-mail**

**Phone**

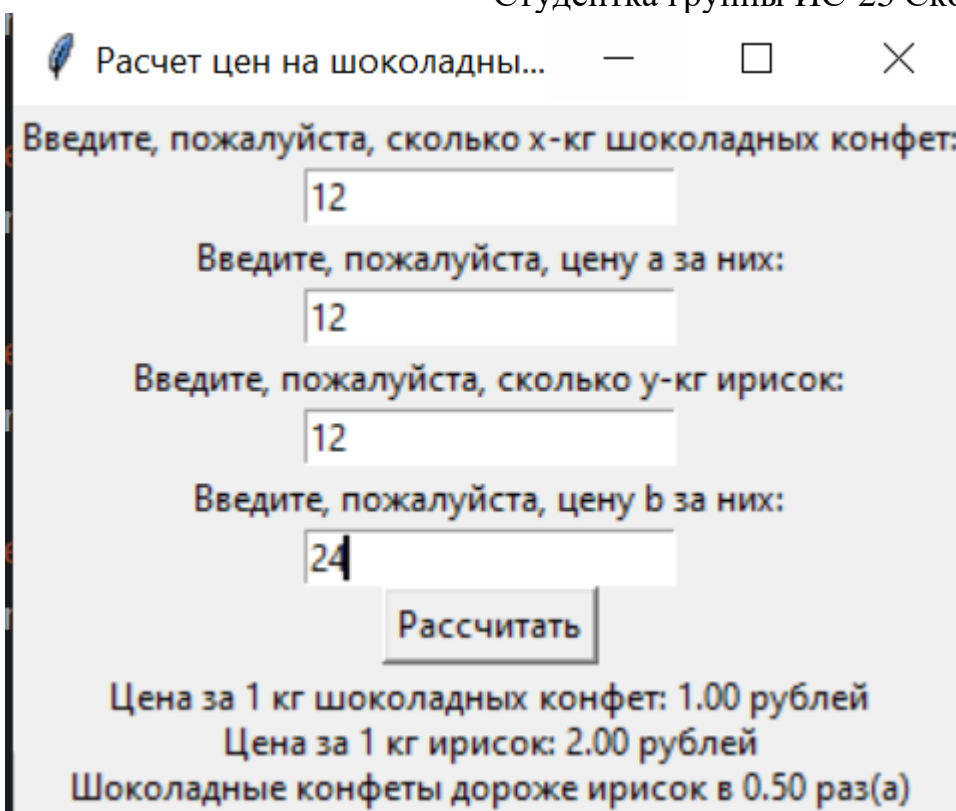
**Password**

**Confirm Password**

☐ I agree to the Terms of Use

**Cancel** **Submit**

1. Process finished with exit code 0  
Программа успешно завершена!

2. 

Process finished with exit code 0

Программа успешно завершена!

**Вывод:** в процессе выполнения практического занятия были выработаны навыки составления программ с использованием GUI Tkinter в IDE PyCharm Community, изучены возможности модуля OS и закреплены усвоенные знания, понятия, алгоритмы, основные принципы составления программ. Выполнены разработка кода, отладка, тестирование, оптимизация программного кода. Готовые программные коды выложены на GitHub.