Министерство образования Республики Беларусь Белорусский Национальный Технический Университет

Факультет информационных технологий и робототехники

Кафедра «Программное обеспечение информационных систем и технологий»

**Отчёт**

по лабораторной работе № 15 по дисциплине ***«Языки программирования»***

тема: **«Разработка игры “My PySneak”»**

Выполнил: студент группы 10701218

Пугачёв В.В.

Принял: ст. пр. Лапанович И.О.

2018-2019 учебный год

**ЛАБОРАТОРНАЯ РАБОТА № 15**

**Разработка игры “ My PySneak ”**

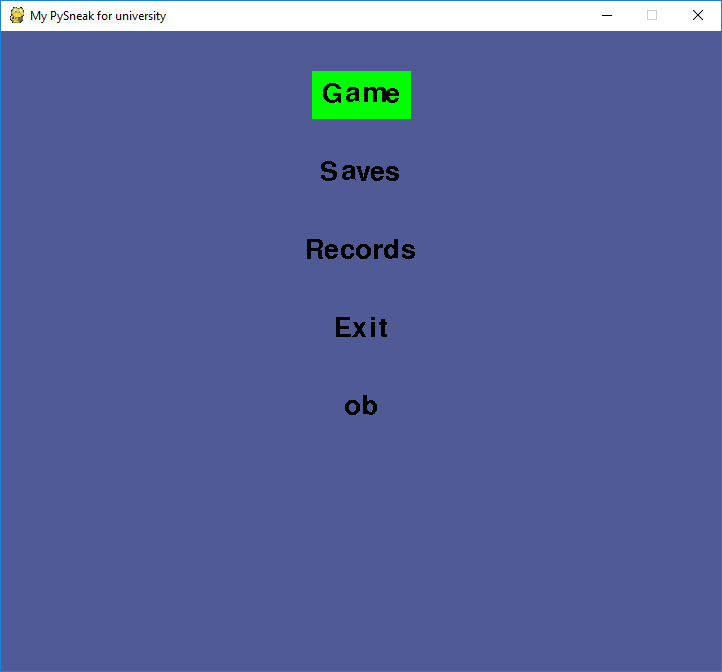
**Цель работы:**

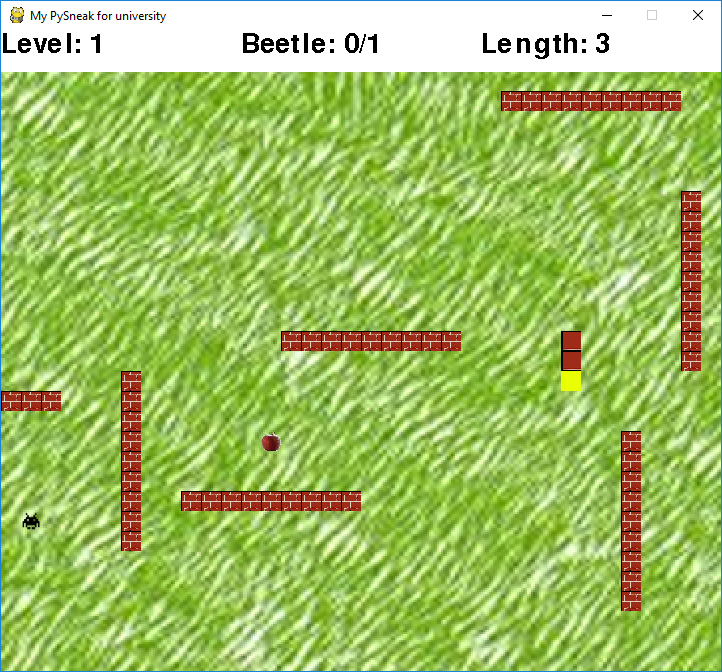
Овладеть знаниями по использованию библиотеки pygame

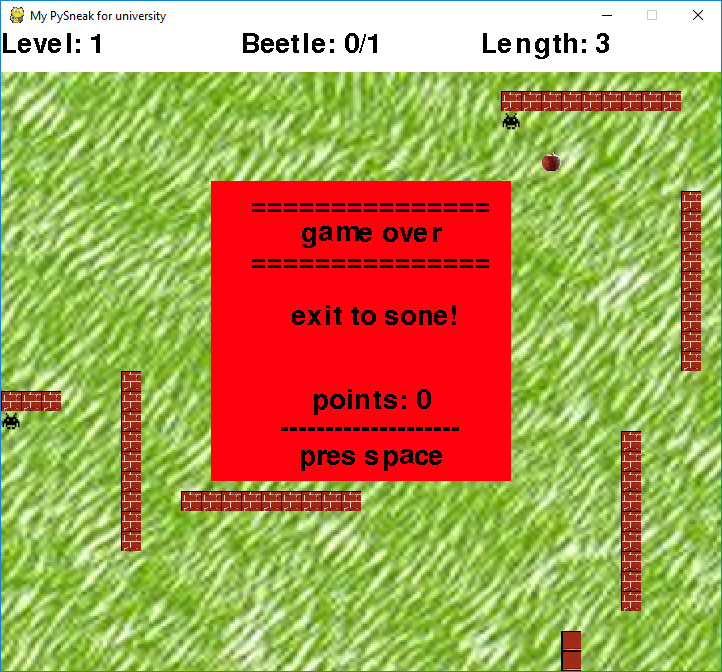
**Основное задание:**

Разработка игры “ **My PySneak** ”

**Результаты выполнения индивидуального задания:**







*ПРИЛОЖЕНИЕ A*

**Листинг исходных кодов.**

if \_\_name\_\_=="\_\_main\_\_":

menu()

def menu():

global GAME, LEVEL

LEVEL = 1

PG.init()

PG.font.init()

Surf = PG.display.set\_mode(ALL\_SIZE["body"])

PG.display.set\_caption(ALL\_TEXT["title"])

try:

Font = PG.font.SysFont('UbuntuMono-R.ttf', SIZEx \* 20)

except:

Font = PG.font.SysFont(None, SIZEx \* 20)

print("load font faled")

try:

bild\_sneak = PG.image.load(ALL\_FILE["bild\_sneak"])

bild\_sneak = PG.transform.scale(bild\_sneak, ALL\_SIZE["segment"])

ALL\_BILD["bild\_sneak"] = bild\_sneak.convert()

except:

print("load sneak\_bild faled")

pass

try:

bild\_sneak\_head = PG.image.load(ALL\_FILE["bild\_sneak\_head"])

bild\_sneak\_head = PG.transform.scale(bild\_sneak\_head, ALL\_SIZE["segment"])

ALL\_BILD["bild\_sneak\_head"] = bild\_sneak\_head.convert()

except:

print("load bild\_sneak\_head faled")

pass

try:

bild\_beetle = PG.image.load(ALL\_FILE["bild\_beetle"])

bild\_beetle = PG.transform.scale(bild\_beetle, ALL\_SIZE["segment"])

ALL\_BILD["bild\_beetle"] = bild\_beetle#.convert()

except:

print("load beetle\_bild faled")

pass

try:

bild\_wall = PG.image.load(ALL\_FILE["bild\_wall"])

bild\_wall = PG.transform.scale(bild\_wall, ALL\_SIZE["segment"])

ALL\_BILD["bild\_wall"] = bild\_wall.convert()

except:

print("load wall\_bild faled")

pass

try:

bild\_eat = PG.image.load(ALL\_FILE["bild\_eat"])

bild\_eat = PG.transform.scale(bild\_eat, ALL\_SIZE["segment"])

ALL\_BILD["bild\_eat"] = bild\_eat

except:

print("load wall\_eat faled")

pass

try:

bild\_place = PG.image.load(ALL\_FILE["bild\_place"])

bild\_place = PG.transform.scale(bild\_place, ALL\_SIZE["plain"])

ALL\_BILD["bild\_place"] = bild\_place.convert()

except:

print("load place\_bild faled")

pass

try: ALL\_SOUN["sound\_eat"] = PG.mixer.Sound(ALL\_FILE["sound\_eat"])

except: print("load sound\_eat faled")

try: ALL\_SOUN["sound\_bittles"] = PG.mixer.Sound(ALL\_FILE["sound\_bittles"])

except: print("load sound\_bittles faled")

try: ALL\_SOUN["sound\_wall"] = PG.mixer.Sound(ALL\_FILE["sound\_wall"])

except: print("load sound\_wall faled")

try: ALL\_SOUN["sound\_exit"] = PG.mixer.Sound(ALL\_FILE["sound\_exit"])

except: print("load sound\_exit faled")

try: ALL\_SOUN["sound\_win"] = PG.mixer.Sound(ALL\_FILE["sound\_win"])

except: print("load sound\_win faled")

try:

PG.mixer.music.load(ALL\_FILE["music\_ground"])

PG.mixer.music.play(-1)

PG.mixer.music.set\_volume(0.1)

except: print("load music\_ground faled")

punkt = 0

Menu\_punkt = []

for i in range(len(ALL\_TEXT["menu"])): Menu\_punkt.append(Font.render(ALL\_TEXT["menu"][i], False, (0, 0, 0)))

menu\_padding = (ALL\_SIZE["body"][1]//len(Menu\_punkt)-Menu\_punkt[0].get\_height())//2

#Surf.blit(Menu\_punkt[i], (ALL\_SIZE["body"][0] // 2 - Menu\_punkt[i].get\_width() // 2, 0 + i \* Menu\_punkt[i].get\_height()))

PG.display.update();

while 1:

Surf.fill(ALL\_COLO["menu"])

for i in PG.event.get():

if i.type == PG.QUIT:

PG.quit()

exit()

return

elif i.type == PG.KEYDOWN and i.key == PG.K\_RETURN:

if ALL\_TEXT["menu"][punkt] == "Game":

try: PG.mixer.music.stop()

except: pass

game(Surf, Font)

try:

PG.mixer.music.load(ALL\_FILE["music\_ground"])

PG.mixer.music.play(-1)

PG.mixer.music.set\_volume(0.1)

except: pass

if ALL\_TEXT["menu"][punkt] == "Saves": saves(Surf, Font)

if ALL\_TEXT["menu"][punkt] == "Records": records(Surf, Font)

if ALL\_TEXT["menu"][punkt] == "ob": autor(Surf, Font)

if ALL\_TEXT["menu"][punkt] == "Exit":

PG.quit()

return

elif i.type == PG.KEYDOWN and i.key == PG.K\_DOWN: punkt += 1

elif i.type == PG.KEYDOWN and i.key == PG.K\_UP: punkt -= 1

else: continue

if QUIT:

PG.quit()

return

if punkt >= len(ALL\_TEXT["menu"]): punkt = 0

if punkt < 0: punkt = len(ALL\_TEXT["menu"]) - 1

for i in range(len(ALL\_TEXT["menu"])):

if i == punkt:

PG.draw.rect(Surf, ALL\_COLO["menu\_shadow"],

(ALL\_SIZE["body"][0] // 2 - Menu\_punkt[i].get\_width() // 2 - 10, menu\_padding + i \*( Menu\_punkt[i].get\_height() + menu\_padding) - ALL\_SIZE["padding"][1]) + (

Menu\_punkt[i].get\_width() + 2\*ALL\_SIZE["padding"][0], Menu\_punkt[i].get\_height() + 2\*ALL\_SIZE["padding"][1]))

Surf.blit(Menu\_punkt[i], (ALL\_SIZE["body"][0] // 2 - Menu\_punkt[i].get\_width() // 2, menu\_padding + i \*( Menu\_punkt[i].get\_height() + menu\_padding)))

PG.display.update()