

ОТЧЕТ

Ошибка 1 — ошибка границы цикла (off-by-one)

Место: src/services/[casino.py](#), функция run_simulation

Симптом: при заданных n шагах симуляция выполняет n+1 шагов

Как воспроизвести: запустить с --steps 1 и --seed 25

Отладка: установить breakpoint на цикл, который крутит шаги симуляции, и посмотреть как правильно считать шаги

Причина: неверно определена граница цикла

Исправление: в объявлении цикла: `while casino.steps_count <= steps` заменить <= на <

Проверка: теперь симуляция выполняет заданное кол-во шагов

Доказательства:

замечаем что в начале атрибут класса Казино steps_count на 1 меньше заданного steps

```
> models
> services
  __init__.py
  main.py
  src_fixed
    collections
      models
        __init__.py
        chips.py
242
243     ##### ОШИБКА 1 - Off-by-one
244     while casino.steps_count <= steps:
245         print()
246         print(f"=====[ШАГ {casino.steps_count + 1}]====")
247         casino.step()
248         print(f"=====[ШАГ {casino.steps_count}]====")
249     print()

Debug main ×
Threads & Variables Console
MainThread Evaluate expression (Enter) or add a watch (Ctrl+Shift+Enter)
run_simulation, casino.py:243
run, main.py:12
wrapper, main.py:691
Invoke, core.py:824
Invoke, core.py:1269
Invoke, core.py:1873
_main, core.py:192
main, core.py:803
__call__, core.py:1485
__call__, main.py:310
<module>, main.py:20
casino = (Casino) <src_buggy.services.casino.Casino object at 0x0000019BFE6AFE00>
balances = {CasinoBalance: 4} {'Саня': 100, 'Вован': 220, 'зхс_иван': 320, 'Иоанн': 65}
gooses = (GooseCollection: 2) GooseCollection(2 гусей)
gooses_income = (GoosesIncomeCollection: 2) {'Васяня': 0, 'Клык': 0}
names = ([list: 24] ['Алеко', 'Боб', 'Виктория', 'Джон', 'Майк', 'Сара', 'Елена', 'Дмитрий', 'Ольга', 'Макс', 'Ник', 'Кен']
players = (PlayerCollection: 4) PlayerCollection(4 игроков)
steps_count = (int) 0
Protected Attributes
seed = (int) 25
steps = (int) 1
```

потом видим, что на последнем шагу steps_count уже на 2 раза увеличился при заданном 1

The screenshot shows the PyCharm debugger interface. The left pane displays the call stack for the MainThread, with the current frame being run_simulation at line 243 of casino.py. The right pane shows the inspection of the 'casino' object. The 'steps_count' attribute is highlighted in blue, indicating it has been evaluated. The value is shown as {int} 2. Other attributes visible include balances, gooses, gooses_income, names, players, seed, and steps.

```
casino = {Casino} <src_buggy.services.casino.Casino object at 0x0000019BFE6AFE00>
    > balances = {CasinoBalance: 4} {'Саня': 196, 'Вован': 220, 'zxc_иван': 320, 'Иоанн': 58}
    > gooses = {GooseCollection: 2} GooseCollection(2 гусей)
    > gooses_income = {GoosesIncomeCollection: 2} {'Вася': 0, 'Клык': 0}
    > names = {list: 24} ['Алекс', 'Боб', 'Виктория', 'Джон', 'Майк', 'Сара', 'Елена', 'Дмитрий', 'Лиза', 'Олег', 'Анна', 'Мария', 'Петр', 'Саша', 'Наташа', 'Юлия', 'Андрей', 'Сергей', 'Денис', 'Андрей', 'Сергей', 'Денис', 'Андрей', 'Сергей', 'Денис']
    > players = {PlayerCollection: 4} PlayerCollection(4 игроков)
        10 steps_count = {int} 2
    > Protected Attributes
        10 seed = {int} 25
        10 steps = {int} 1
```

значит нужно сократить границы цикла

Ошибка 2 — неверное логическое условие

Место: src/services/[casino.py](#), метод panic_action класса Casino

Симптом: если это событие случается, оно правильно выводится в консоль, но под капотом ничего не меняется: игрок не убегает, и рандомный гусь не получает его деньги

Как воспроизвести: запустить симуляцию с –steps 2 и –seed 20

Причина: все, что должно выполняться под капотом, случается при ненулевом балансе игрока, у которого случилась паника, но в условии написано: `if amount < 0` – то есть оно всегда False при ненулевом балансе

Отладка: поставить брейкпоинты на строчки условия и вызова шагов, сначала дойти до шага с выполнением условия, заметить положительный баланс, а потом следить за коллекциями казино, меняется ли там что-то с этим игроком и доходами гусей

Исправление: замена < на > в условии `if amount < 0`

Проверка: Теперь паника пользователя срабатывает корректно под

капотом, пользователь удаляется из всех коллекций казино, а доход рандомно выбранного гуся увеличивается на баланс убежавшего

Доказательства:

замечаем положительный баланс у игрока, который должен будет потом отовсюду удалиться, а его баланс перейти рандомному гусю

The screenshot shows the PyCharm debugger interface. The code editor displays a section of `main.py` with a red dot indicating a break point at line 165. The code is as follows:

```
164
165     if amount < 0:
166         self.goses_income[goose.name] += amount
167
168         print(" - Игрок")
169         del self.balances[player.name]
170         self.players.remove(player)
171
172         msg += f"Гусь {goose.name} забрал со стола оставленные игроком ${amount}
```

The stack trace on the left shows the call stack starting from `panic_action` in `casino.py` at line 166. The current frame is `main` at line 165. The variable evaluation window on the right shows the state of variables:

- `amount = {int} 100`
- `goose = {HonkGoose} HonkGoose Клык с криком 5`
- `msg = {str} 'Игрок Саня запаниковал при виде агрессивно настроенного гуся Клык и убежал в страхе из зала'`
- `player = {Player} Игрок Саня с балансом 100 и здоровьем 100`
- `self = {Casino} <src_buggy.services.casino.Casino object at 0x000001F8B36BBE00>`
 - `balances = {CasinoBalance: 4} {'Саня': 100, 'Вован': 149, 'zxc_иван': 320, 'Иоанн': 65}`
 - `gooses = {GooseCollection: 2} GooseCollection(2 гусей)`
 - `gooses_income = {GoosesIncomeCollection: 2} {'Вася': 0, 'Клык': 0}`
 - `names = {list: 24} ['Алекс', 'Боб', 'Виктория', 'Джон', 'Майк', 'Сара', 'Елена', 'Дмитрий', 'Ольга', 'Макс', 'Ни`
 - `players = {PlayerCollection: 4} PlayerCollection(4 игроков)`
- `steps_count = {int} 2`
- `Protected Attributes`

а потом заметить, что игрок не удалился, так как число игроков коллекции не изменилось, баланс любого гуся не изменился:

```

> collections
> models
> services
  ↘ __init__.py
  ↘ main.py
src_fixed
  > collections
  > models
    ↘ __init__.py
    ↘ chips.py

```

```

243     while casino.steps_count <= steps:
244         print()
245         print(f"———— [ШАГ {casino.steps_count + 1}]")
246         casino.step()
247         print(f"———— [ШАГ {casino.steps_count}]")
248
249     print()

```

Debug main x

MainThread

- run_simulation, casino.py:246
- run, main.py:12
- wrapper, main.py:691
- invoke, core.py:824
- invoke, core.py:1269
- invoke, core.py:1873
- _main, core.py:192
- main, core.py:803
- __call__, core.py:1485
- __call__, main.py:310
- <module>, main.py:20

Evaluate expression (Enter) or add a watch (Ctrl+Shift+Enter)

```

casino = {Casino} <src_buggy.services.casino.Casino object at 0x0000014BBAE43B60>
> balances = {CasinoBalance: 4} {'Саня': 100, 'Вован': 149, 'хс_иван': 320, 'Иоанн': 65}
> gooses = {GooseCollection: 2} GooseCollection(2 гусей)
> gooses_income = {GoosesIncomeCollection: 2} {'Васяня': 0, 'Клык': 0}
  > data = {dict: 2} {'Васяня': 0, 'Клык': 0}
    > 'Васяня' = {int} 0
    > 'Клык' = {int} 0
    > __len__ = {int} 2
      > Protected Attributes
      > Protected Attributes
  > names = {list: 24} ['Алекс', 'Боб', 'Виктория', 'Джон', 'Майк', 'Сара', 'Елена', 'Дмитрий', 'Ольга', ...]
  > players = {PlayerCollection: 4} PlayerCollection(4 игроков)
  > steps_count = {int} 2

```

и пойти исправлять условие с amount

Ошибка 3 — сравнение через is вместо ==

Место: src/services/[casino.py](#), метод register_player класса Casino

Симптом: если в симуляции выпадает регистрация игрока, я в сигнатуре не указываю его баланс, поэтому в методе ставится дефолтное значение, а потом если срабатывает условие проверки на дефолтное - баланс выбирается рандомный

Как воспроизвести: запустить с –steps 15 и –seed 20, но сработает оно так же правильно как и с ==

Отладка: поставил брейкпоинт на условие, запустил отладчик

Причина: проверка совпадения с дефолтом: `if balance is -1`

При таком условии проверка строк становится ненадежной потому-что при is значения могут и совпадать, но они могут находиться в

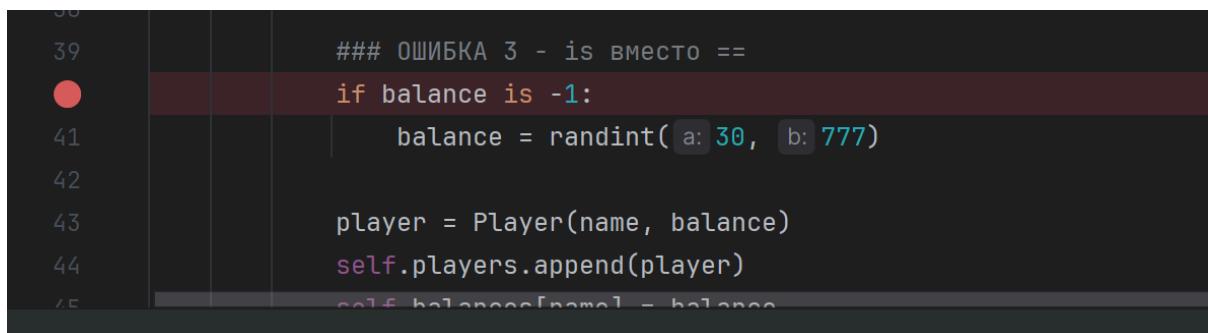
разных участках памяти, а `is` как раз и проверяет, являются ли они одним объектом в памяти

Исправление: заменить `is` на `==`

Проверка: теперь игрок случайный зарегистрируется без проблем, потому что нам главное проверять значения, то есть использовать `==`

Доказательства:

точка останова:



```
39     ##### ОШИБКА 3 - is вместо ==
40     ●
41         if balance is -1:
42             balance = randint(a: 30, b: 777)
43
44         player = Player(name, balance)
45         self.players.append(player)
46         self.balance[player] = balance
```

я хоть и пытался поймать неправильную работу, все же программа работала правильно, так же как и с `==`, но это все же не всегда так возможно. и IDE тоже об этом предупреждает:

```
import sys; print('Python %s on %s' % (sys.version, sys.platform))
C:\Users\izoto\PycharmProjects\mai-python-lab5-debug\.venv\Scripts\python.exe -X pycache_prefix=C:\Users\izoto\AppData\Local\JetBrains\PyCharmCE2024.3\c
Connected to pydev debugger (build 243.26053.29)
C:\Users\izoto\PycharmProjects\mai-python-lab5-debug\src_buggy\services\casino.py:40: SyntaxWarning: "is" with 'int' literal. Did you mean "=="?
  if balance is -1:
    + Игрок
    + Игрок
    + Игрок
    + Игрок
    + Гусь
    + Гусь
>>> |
```

Ошибка 4 — перехват слишком общего исключения

Место: `src/services/casino.py`, метод `step` класса `Casino`

Симптом: при возникновении какой-либо ошибки в симуляции программа, когда вызывает случайное событие через `try-except`, скрывает классифицированную ошибку, ловит просто `Exception`

Как воспроизвести: запустить с `-steps 1` и `-seed 22`

Отладка: breakpoint в строке с принтом внутри `except Exception as e:`, поиск настоящих исключений

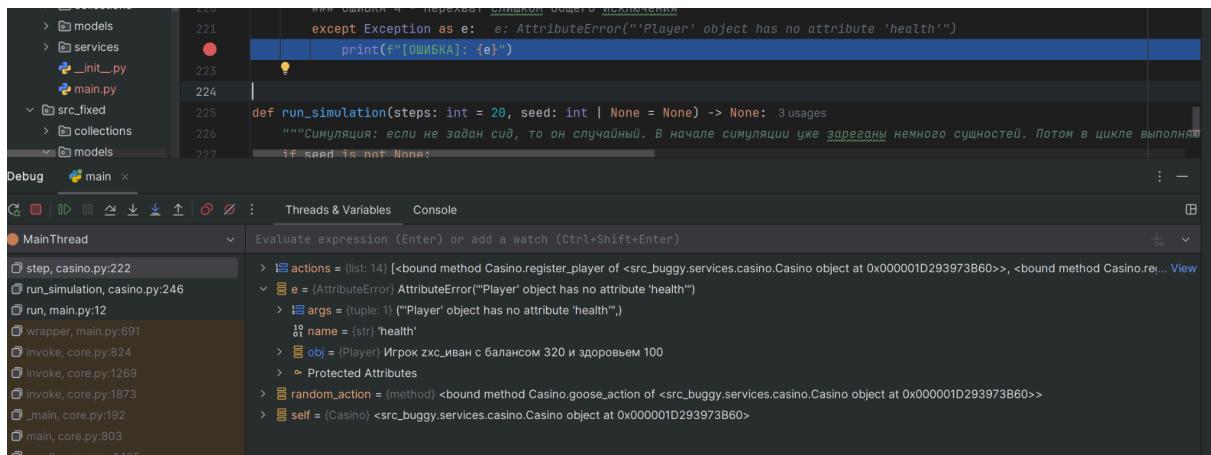
Причина: `except Exception as e: print(f"[ОШИБКА]: {e}")` перехватывает все исключения, которые он охватывает, при этом скрывает информацию об ошибке, необходимую об отладке

Исправление: легче всего улучшить print:

```
print(f"[ОШИБКА] {type(e).__name__}: {e}")
```

Проверка: теперь при возникновении ошибки, в консоли выводится настоящее исключение на определенном шаге, программа все так же не прерывается

Доказательства: вот случай когда возникает `AttributeError` (поговорим об этом исключении в следующей ошибке), понимаем, что мы проглатываем классификацию исключения



The screenshot shows the PyCharm IDE interface. The code editor has a breakpoint set on line 223. The code on line 223 is:

```
except Exception as e: e: AttributeError("Player' object has no attribute 'health'")
```

The code on line 224 is:

```
print(f"[ОШИБКА]: {e}")
```

The code on line 225 is:

```
def run_simulation(steps: int = 20, seed: int | None = None) -> None: 3 usages
```

The code on line 226 is a multi-line string:

```
"""Симуляция: если не задан seed, то он случайный. В начале симуляции уже зареганы немногие существа. Потом в цикле выполняется"""
if seed is not None:
```

The bottom part of the screenshot shows the 'Threads & Variables' tool window. It lists several frames, with the top one being:

```
MainThread
```

Below it, the stack trace starts with:

```
step, casino.py:222
```

Then it shows the exception frame:

```
run_simulation, casino.py:246
```

And continues with:

```
run, main.py:12
```

Further down the stack trace are:

```
wrapper, main.py:691
```

```
invoke, core.py:824
```

```
invoke, core.py:1269
```

```
invoke, core.py:1873
```

```
_main, core.py:192
```

```
main, core.py:803
```

Ошибка 5 — обращение к несуществующему атрибуту объекта

Место: `src/models/gooses.py`, метод `action` класса `WarGoose`

Симптом: при выполнении действия боевого гуся, он взаимодействует в поле здоровья у игрока, вылетает `AttributeError`

Как воспроизвести: запустить с `-steps 1` и `-seed 22`

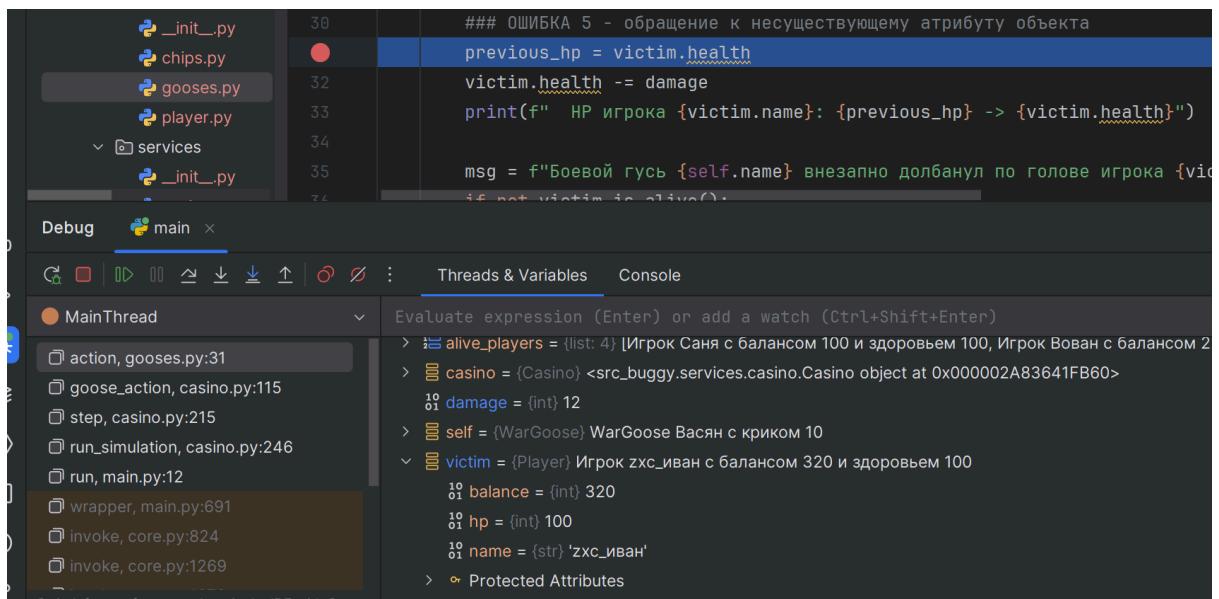
Отладка: установить breakpoint на первую строчку, в которой идет обращение к атрибуту, заметить что существует атрибут здоровья с другим названием

Причина: в методе гуся обращение идет по не существующему атрибуту здоровья объекта игрока: `victim.health`

Исправление: указать правильный атрибут здоровья везде:
`victim.health -> victim.hp`

Проверка: теперь взаимодействие происходит с существующим атрибутом, здоровье экземпляра игрока правильно меняется и ошибок не вылетает никаких

Доказательства:



ОШИБКА 5 - обращение к несуществующему атрибуту объекта
previous_hp = victim.health
victim.health -= damage
print(f" HP игрока {victim.name}: {previous_hp} -> {victim.health}")

msg = f"Боевой гусь {self.name} внезапно долбанул по голове игрока {victim.name}!"
if not victim.is_alive():

MainThread
action, gooses.py:31
goose_action, casino.py:115
step, casino.py:215
run_simulation, casino.py:246
run, main.py:12
wrapper, main.py:691
invoke, core.py:824
invoke, core.py:1269

Evaluate expression (Enter) or add a watch (Ctrl+Shift+Enter)
> alive_players = [Player] Игрок Саня с балансом 100 и здоровьем 100, Игрок Вован с балансом 200 и здоровьем 100, Игрок Коля с балансом 300 и здоровьем 100, Игрок Вася с балансом 400 и здоровьем 100
> casino = Casino <src_buggy.services.casino.Casino object at 0x000002A83641FB60>
> damage = 12
> self = WarGoose WarGoose Вася с криком 10
> victim = Player Игрок zxc_иван с балансом 320 и здоровьем 100
> victim.balance = 320
> victim.hp = 100
> victim.name = 'zxc_иван'
> Protected Attributes