Calcularea ariei cercului în Python

Tutorial pas cu pas pentru realizarea și înțelegerea programului în PyCharm

Ce veți învăța

Programul solicită raza unui cerc, calculează aria și afișează rezultatul cu două zecimale. Pe parcurs sunt explicate funcțiile, citirea datelor, modulul math, f-stringurile, formatarea numerelor, rularea în PyCharm și mesajul slowdown.

- crearea și rularea unui program în PyCharm;
- citirea unei valori cu input și transformarea ei cu float;
- definirea și apelarea unei funcții;
- folosirea valorii math.pi;
- construirea unui mesaj cu un f-string;
- afișarea unui număr cu două zecimale.

Formula matematică

Aria cercului se calculează cu formula $A = \pi \times r^2$, unde A este aria, r este raza, iar π este numărul pi, aproximativ 3,141592.

Programul complet

```
import math


def aria_cercului(r):
    return math.pi * r * r


def main():
    r = float(input("r = "))
    print(f"Aria este {aria_cercului(r):.2f}")


if __name__ == "__main__":
    main()
```

Atenție: Python folosește indentarea pentru a arăta ce instrucțiuni aparțin unei funcții sau unei structuri. În PyCharm, indentarea este realizată de regulă cu patru spații.

Crearea programului în PyCharm

1. Deschideți PyCharm și proiectul PrimulProiect.

2. În panoul Project, dați dublu clic pe fișierul main.py.

3. Introduceți programul complet de mai sus.

4. Apăsați Ctrl + S pentru salvare manuală. PyCharm salvează, de regulă, și automat.

Explicarea programului

Importarea modulului math

```
import math
```

Instrucțiunea încarcă modulul math, o colecție de instrumente matematice deja definite. Programul utilizează math.pi, o valoare precisă a numărului π. Expresia math.pi se poate citi: «din modulul math folosim valoarea pi».

Definirea funcției aria_cercului

```
def aria_cercului(r):
    return math.pi * r * r
```

Cuvântul-cheie def definește o funcție. Numele funcției este aria_cercului, iar r este parametrul care primește raza. Caracterul două puncte marchează începutul corpului funcției.

Instrucțiunea return calculează π × r × r și trimite rezultatul înapoi la locul din care funcția a fost apelată. Simbolul * reprezintă înmulțirea. Funcția calculează rezultatul, dar nu îl afișează singură.

Definirea funcției principale

```
def main():
```

Funcția main conține acțiunile principale: citirea razei, apelarea funcției de calcul și afișarea rezultatului. Parantezele sunt goale deoarece main nu primește parametri.

Citirea razei

```
r = float(input("r = "))
```

Instrucțiunea se execută din interior spre exterior:

1. input("r = ") afișează mesajul r = și așteaptă ca utilizatorul să introducă o valoare;

2. input returnează întotdeauna text, chiar dacă utilizatorul tastează cifre;

3. float transformă textul într-un număr real;

4. valoarea numerică este memorată în variabila r.

Pentru numere zecimale se folosește punctul: 2.5, nu 2,5.

Afișarea rezultatului

```
print(f"Aria este {aria_cercului(r):.2f}")
```

Funcția print afișează mesajul. În interior se află un f-string, adică un șir de caractere formatat.

Primul f și rolul f-stringului

Litera f dinaintea ghilimelelor vine de la formatted string. Ea îi spune lui Python că în text există valori sau expresii între acolade care trebuie evaluate și introduse în mesaj.

```
nume = "Ana"
print(f"Bună, {nume}!")
```

Rezultatul este: Bună, Ana! Python înlocuiește {nume} cu valoarea variabilei.

Fără litera f, instrucțiunea următoare ar afișa acoladele și numele variabilei exact așa cum sunt scrise:

```
nume = "Ana"
print("Bună, {nume}!")
```

Rezultatul ar fi: Bună, {nume}!

Între acoladele unui f-string poate apărea o variabilă, un calcul sau apelul unei funcții. În programul cercului, aria_cercului(r) este apelată, iar rezultatul returnat este introdus în mesaj.

Formatarea cu două zecimale

```
{aria_cercului(r):.2f}
```

Aici, semnul : introduce regulile de formatare, punctul introduce precizia, cifra 2 solicită două cifre după punctul zecimal, iar ultimul f cere afișarea ca număr real în format zecimal.

Cele două litere f au roluri diferite: primul f, aflat înaintea ghilimelelor, creează f-stringul; ultimul f, din :.2f, formatează numărul. Valoarea 314.159265... este rotunjită și afișată 314.16.

Pornirea funcției main

```
if __name__ == "__main__":
    main()
```

Python creează automat variabila specială __name__. Când fișierul este executat direct, valoarea acesteia este "__main__". Condiția devine adevărată și este apelată funcția main. Această organizare este utilă mai ales în proiecte mai mari și reprezintă o practică bună.

De ce apar caractere underscore

În __name__ și __main__ apar câte două caractere underscore înainte și după cuvânt. Aceste nume se citesc adesea «dunder name» și «dunder main». Termenul dunder vine de la double underscore, adică underscore dublu.

__name__ este o variabilă specială creată automat de Python pentru fiecare fișier. Nu noi alegem acest nume și nu trebuie să îl modificăm. "__main__" este un șir de caractere special, nu o variabilă, deoarece este scris între ghilimele.

Când un fișier este rulat direct, Python îi atribuie variabilei __name__ valoarea "__main__". De aceea condiția devine adevărată și se execută main(). Numele cu underscore dublu la început și la sfârșit sunt folosite de Python pentru elemente cu semnificație specială.

În numele aria_cercului apare un singur underscore între cuvinte. Acesta înlocuiește spațiul, deoarece numele unei funcții nu poate conține spații. Prin urmare, aria cercului este incorect ca nume de funcție, iar aria_cercului este corect și ușor de citit.

Varianta simplificată pentru începători

Într-un program mic putem evita atât funcția main(), cât și verificarea care folosește __name__. Programul poate fi scris astfel:

```
import math


def aria_cercului(r):
    return math.pi * r * r


r = float(input("r = "))
print(f"Aria este {aria_cercului(r):.2f}")
```

Această variantă funcționează la fel și este potrivită pentru primele lecții de Python. Ea importă modulul math, definește funcția, citește raza și afișează aria.

Construcția cu main() și if __name__ == "__main__": devine utilă mai târziu, când programul este mai mare, conține mai multe fișiere sau când dorim să importăm funcții dintr-un fișier în altul. Pentru elevii aflați la început poate fi folosită mai întâi varianta simplificată.

Ordinea executării

1. Se importă modulul math.

2. Se memorează definiția funcției aria_cercului.

3. Se memorează definiția funcției main.

4. Se verifică dacă fișierul este executat direct.

5. Se apelează main().

6. Se afișează r = și se citește raza.

7. Textul introdus este transformat în număr real.

8. Se apelează aria_cercului(r).

9. Se calculează și se returnează aria.

10. Rezultatul este rotunjit la două zecimale și afișat.

Rularea programului

1. Dați clic dreapta în zona codului și selectați Run 'main'.

2. În fereastra Run, la mesajul r =, introduceți raza, de exemplu 10.

3. Apăsați Enter.

```
r = 10
Aria este 314.16

Process finished with exit code 0
```

Mesajul Process finished with exit code 0 confirmă că programul s-a terminat fără erori.

Exemple

Raza	Calcul	Rezultat afișat
5	π × 5 × 5	Aria este 78.54
2.5	π × 2.5 × 2.5	Aria este 19.63
10	π × 10 × 10	Aria este 314.16

Greșeli frecvente

Virgulă în loc de punct: Scrierea 2,5 produce ValueError. Introduceți 2.5.

Text în loc de număr: Cuvântul zece nu poate fi transformat cu float. Introduceți 10.

Indentare greșită: Instrucțiunile din funcții și instrucțiunea main() din if trebuie deplasate spre dreapta.

Ghilimele sau două puncte lipsă: Textele trebuie puse între ghilimele, iar definițiile funcțiilor și condiția if se termină cu două puncte.

Mesajul PyCharm has encountered a slowdown

Acest mesaj nu este o eroare a programului Python. PyCharm a observat doar că o operație internă a răspuns mai lent decât era normal. Se poate întâmpla în timpul analizării proiectului, configurării mediului .venv, indexării bibliotecilor, verificării fișierelor de către Microsoft Defender sau atunci când calculatorul rulează simultan mai multe programe.

Dacă în fereastra Run apare Process finished with exit code 0, programul Python a fost executat corect.

- Dismiss închide avertismentul;
- Mute problem ascunde avertismentele similare;
- Report problem trimite informații tehnice către JetBrains.

Pentru început este suficient să apăsați Dismiss. Dacă avertismentul apare frecvent și PyCharm funcționează greu, închideți și redeschideți aplicația, așteptați terminarea indexării și închideți programele care consumă multe resurse.

Salvarea și închiderea

PyCharm salvează de obicei automat. Pentru salvare manuală, apăsați Ctrl + S sau alegeți File și apoi Save All.

```
C:\Users\lilis\PycharmProjects\PrimulProiect
main.py
```

Pentru închidere, apăsați X în colțul din dreapta sus. La următoarea pornire, selectați PrimulProiect din lista proiectelor recente și deschideți main.py. Folderul .venv este administrat de PyCharm și nu trebuie modificat sau șters.

Noțiuni învățate

- modulul math și valoarea math.pi;
- funcții, parametri și return;
- funcția principală main;
- citirea datelor cu input;
- transformarea datelor cu float;
- variabile și apeluri de funcții;
- f-stringuri și expresii între acolade;
- formatarea la două zecimale;
- rularea, salvarea și interpretarea rezultatului în PyCharm.

Deși este scurt, programul este bine structurat și reunește mai multe concepte fundamentale din Python.