

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
Національний аерокосмічний університет  
«Харківський авіаційний інститут»

Факультет систем управління літальних апаратів  
Кафедра систем управління літальних апаратів

**Лабораторна робота № 5**

з дисципліни «Алгоритмізація та програмування»  
на тему «Реалізація циклічних алгоритмів мовою C ++»

XAI.301. 16 ЛР

Виконав студент гр. \_\_\_\_\_319\_\_\_\_\_

\_\_\_\_\_Павло Сафонов\_\_\_\_\_

(підпис, дата)

(П.І.Б.)

Перевірив

\_\_\_\_\_к.т.н., доц. Олена ГАВРИЛЕНКО

(підпис, дата)

(П.І.Б.)

## МЕТА РОБОТИ

Вивчити теоретичний матеріал із синтаксису мовою C++ і поданням у вигляді UML діаграм циклічних алгоритмів і реалізувати алгоритми з використанням інструкцій циклу з передумовою, циклу з післяумовою і параметризованого циклу мовою C++ в середовищі Visual Studio.

## ПОСТАНОВКА ЗАДАЧІ

Завдання 1. Дано дійсні числа  $(x_i, y_i)$ ,  $i = 1, 2, \dots, n$ , – координати точок на площині. Визначити кількість точок, що потрапляють в фігуру заданого кольору (або групу фігур). Варіанти фігур представлено в табл.1.

Завдання 2. Дано дійсне число  $x$  і натуральне число  $n$ . Необхідно:

а) Обчислити значення виразу при заданих  $x$  і  $n$  для виразу з табл.2.

б) Вивести: для парних варіантів – значення кожного третього елемента, для непарних – значення кожного четвертого елемента.

Завдання 3. Дослідити ряд на збіжність. Умова закінчення циклу обчислення суми прийняти у вигляді:  $|u_n| < \epsilon$  або  $|u_n| > g$ , де  $\epsilon$  – мала величина

для переривання циклу обчислення суми збіжного ряду  
( $\epsilon = 10^{-5}$

...  $10^{-20}$ );  $g$  – величина для переривання циклу обчислення суми

розбіжного ряду ( $g = 10^2$

...  $10^5$

). Варіанти представлено в табл.3.

Завдання 4. Організувати меню в командному вікні для багаторазового виконання завдань. \*Додати функцію/функції, що вводять з консолі та повертають коректне значення цілого/дійсного типу у відповідності з обмеженнями вхідних даних кожного завдання.

Завдання 5. Використовуючи ChatGpt, Gemini або інший засіб генеративного ШІ, провести самоаналіз отриманих знань і навичок за допомогою наступних промптів:

1) «Ти - викладач, що приймає захист моєї роботи. Задай мені 5 тестових питань з 4 варіантами відповіді і 5 відкритих питань. Це мають бути завдання <середнього> рівня складності на розвиток критичного та

інженерного мислення. Питання мають відноситись до коду, що є у файлі звіту, і до теоретичних відомостей, що є у файлі лекції»

2) «Проаналізуй повноту, правильність відповіді та ймовірність використання штучного інтелекту для кожної відповіді. Оціни кожне питання

у 5-бальній шкалі, віднімаючи 60% балів там, де ймовірність відповіді з засобом ШІ висока. Обчисли загальну середню оцінку»

3) «Проаналізуй код у звіті, і додай опис і приклади коду з питань, які є в теоретичних відомостях, але не відпрацьовано у коді при вирішенні завдань»

Проаналізуйте задані питання, коментарі і оцінки, надані ШІ. Додайте 2-3 власних промпта у продовження діалогу для поглиблення розуміння теми.

Всі завдання повинні бути реалізовані в одному командному додатку. Кожне завдання має бути реалізовано у вигляді окремої процедури (функції без параметрів).

Рекомендації до виконання третього завдання:

1. При налагодженні програми діапазон (e, g) потрібно приймати невеликим, наприклад, (10-2, 102).

2. Наочно аналіз процесу збіжності можна показати на екрані, виконавши висновок значень наростаючою суми з фіксованою позицією точки і з 15 знаками після точки.

3. Якщо процес збіжності повільний, то висновок даних на екран потрібно виконувати з заданим кроком M.

У звіті повинен бути лістинг коду програми та скріншот екрану виконання. Код повинен містити КОМЕНТАРІ !!!

Крім того, в звіті повинно бути представлено чотири діаграми: для всієї програми і для функцій, що реалізують завдання 1,2 і 3.

27  
-31


$r$  - радіус кола

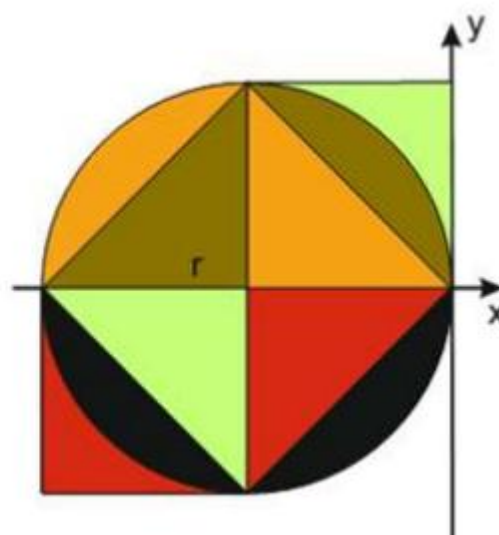
 27 вар.

 28 вар.

 29 вар.

 30 вар.

 31 вар.



19

$$\sum_{k=0}^n \frac{(-1)^k \left(-\frac{\pi}{2} + x\right)^{1+2k}}{(1+2k)!}$$

32

$$\sum_{n=1}^{\infty} \frac{5^n}{\sqrt{n} 2^{n-1}}$$

ВИКОНАННЯ РОБОТИ

Код

```

#include <iostream>
#include <vector>
#include <cmath>

using namespace std;

void Figure(); // task 1

void EnterConsole(vector<double>& pointsX, vector<double>& pointsY, int& N, double& r); // за допомогою цієї методу користувач вводить у консоль для першого завдання
int AmountPointsInFigure(vector<double> pointsX, vector<double> pointsY, int N, double r); // цей метод виконує перше завдання
void Console_Figure(int amount); // цей метод виводить результат пршого завдання у консоль

bool isPointInSector(double x, double y, double r); // цей метод перевіряє чи точка у секторі?
bool isAngleInSector(double angleDeg, double startDeg, double endDeg); // цей метод перевіряє два сектора

void Formula19(); // task 2

void EnterConsole_Formula19(double& x, int& n); // за допомогою цієї методу користувач вводить у консоль для другого завдання
void Formula19Calculation(double& sum, double x, int n); // цей метод виконує друге завдання
void Console_Formula19(double sum); // цей метод виводить результат другого завдання у консоль

void Formula32(); // task 3

void EnterConsole_Formula32(double& e, double& g); // за допомогою цієї методу користувач вводить у консоль для третього завдання
double SumSeries(double e, double g); // цей метод виконує третє завдання
void Console_Formula32(double sum); // цей метод виводить результат третього завдання у консоль

int main()
{
    int menu; // змінна за допомогою якого буде перевірятися вибір користувача у меню

    do // цей цикл використовується як меню
    {
        cin >> menu; // кирустувач обирає завдання

        switch (menu)
        {
            case 1: Figure(); break; // перше
            case 2: Formula19(); break; // друге
            case 3: Formula32(); break; // третє
            case 0: cout << "Bye :)" << endl; break; // вихіді
            default: cout << "only 1, 2 or 3" << endl; break; // користувач захотів обрати неіснуюче завдання
        }
    } while (menu != 0); // нуль це вихід
}

```

```

44     } while (menu != 0); // нуль це вихід
45
46
47     // програма завершається
48     system("pause");
49     return 0;
50 }
51
52 void Figure()
53 {
54     int N; // змінна яка виконує роль розміру для масивів а точніше кількість точок
55
56     double r; // змінна радіус
57
58     vector<double> pointsX; // ось x точки
59     vector<double> pointsY; // ось y точки
60
61     EnterConsole(pointsX, pointsY, N, r); // ці методи вже описані вище
62     int amountPointsInFigure = AmountPointsInFigure(pointsX, pointsY, N, r);
63     Console_Figure(amountPointsInFigure);
64 }
65
66 void EnterConsole(vector<double>& pointsX, vector<double>& pointsY, int& N, double& r)
67 {
68     cout << "Enter n count points: "; // просимо користувача ввести розмір для масивів
69     cin >> N; // користува обирає розмір
70
71     pointsX.resize(N); // міняємо розмір масивів згідно з вибором користувача а точніше кількість точок
72     pointsY.resize(N);
73
74     cout << "Enter radius: "; // просимо користувача ввести радіус
75     cin >> r; // користува вводить радіус
76
77     cout << "Enter x and y points"; // просимо користувача ввести осі
78
79     for (int i = 0; i < N; i++) // цикл для зручності а також працює згідно з кількістю точок
80     {
81         cout << "begin x than y" << endl; // просимо користувача ввести спочатку x а вже потім y
82         cin >> pointsX.at(i) >> pointsY.at(i); // користува вводить осі
83     }
84 }
85

```

```

85
86  ✓ int AmountPointsInFigure(vector<double> pointsX, vector<double> pointsY, int N, double r)
87  {
88      int amount = 0; // змінна в якій буде зберігатися кількість точок які входять у сектора
89
90      for (int i = 0; i < N; i++) // цикл згідно з кількістю точок
91      {
92          amount += isPointInSector(pointsX.at(i), pointsY.at(i), r) ? 1 : 0; // програма перевіряє чи точка у секторі?
93      }
94
95      return amount; // повертаємо цю кількість
96  }
97
98  void Console_Figure(int amount)
99  {
100      cout << "Amount points in figure: " << amount << endl; // виводимо кількість точок у фігурах або секторах
101  }
102
103  // Основная функция проверки
104  ✓ bool isPointInSector(double x, double y, double r)
105  {
106      double distance = sqrt(x * x + y * y); // склонуємо
107      if (distance > r) return false; // не потрапили у радіус значить точно вне будь якої фігури або сектораі
108
109      double angleRad = atan2(y, x);
110      double angleDeg = angleRad * 180.0 / M_PI;
111      if (angleDeg < 0) angleDeg += 360.0;
112
113      // перевірка двох секторів
114      return isAngleInSector(angleDeg, 45.0, 90.0) || isAngleInSector(angleDeg, 90.0, 135.0);
115  }
116
117  bool isAngleInSector(double angleDeg, double startDeg, double endDeg)
118  {
119      return angleDeg >= startDeg && angleDeg <= endDeg; // повертає чи воно у секторах?
120  }
121
122  ✓ void Formula19()
123  {
124      double sum = 0.0; // змінна сума
125
126      double x; // змінна x

```

```

126     double x; // змінна x
127     int n; // змінна n
128
129     EnterConsole_Formula19(x, n); // ці методи описані вище
130     Formula19Calculation(sum, x, n);
131     Console_Formula19(sum);
132 }
133
134 void EnterConsole_Formula19(double& x, int& n)
135 {
136     cout << "Enter x and n" << endl; // просимо користувача ввести x а потім n
137     cin >> x >> n; // користувач вводить
138 }
139
140 void Formula19Calculation(double& sum, double x, int n)
141 {
142     for (int k = 0; k <= n; k++) // цикл згідно формули
143     {
144         double element = pow(-1, k) * pow(-M_PI/2+x, 1+2*k)/tgamma(1+2*k+1); // змінна зберігає елемент
145
146         sum += element; // додаємо елемент до загальної суми
147
148         if (k % 4 == 0) // кожному четвертий елемент треба вивести у консоль
149             cout << "Element k-" << k+1 << " : " << element << endl;
150     }
151 }
152
153 void Console_Formula19(double sum)
154 {
155     cout << "your sum: " << sum << endl; // виводимо суму
156 }
157
158 void Formula32()
159 {
160     double e, g; // змінні мала та велика величина
161
162     EnterConsole_Formula32(e, g); // ці методи описані вище
163     double sum = SumSeries(e, g);
164     Console_Formula32(sum);
165 }
166
167 void EnterConsole_Formula32(double& e, double& g)

```

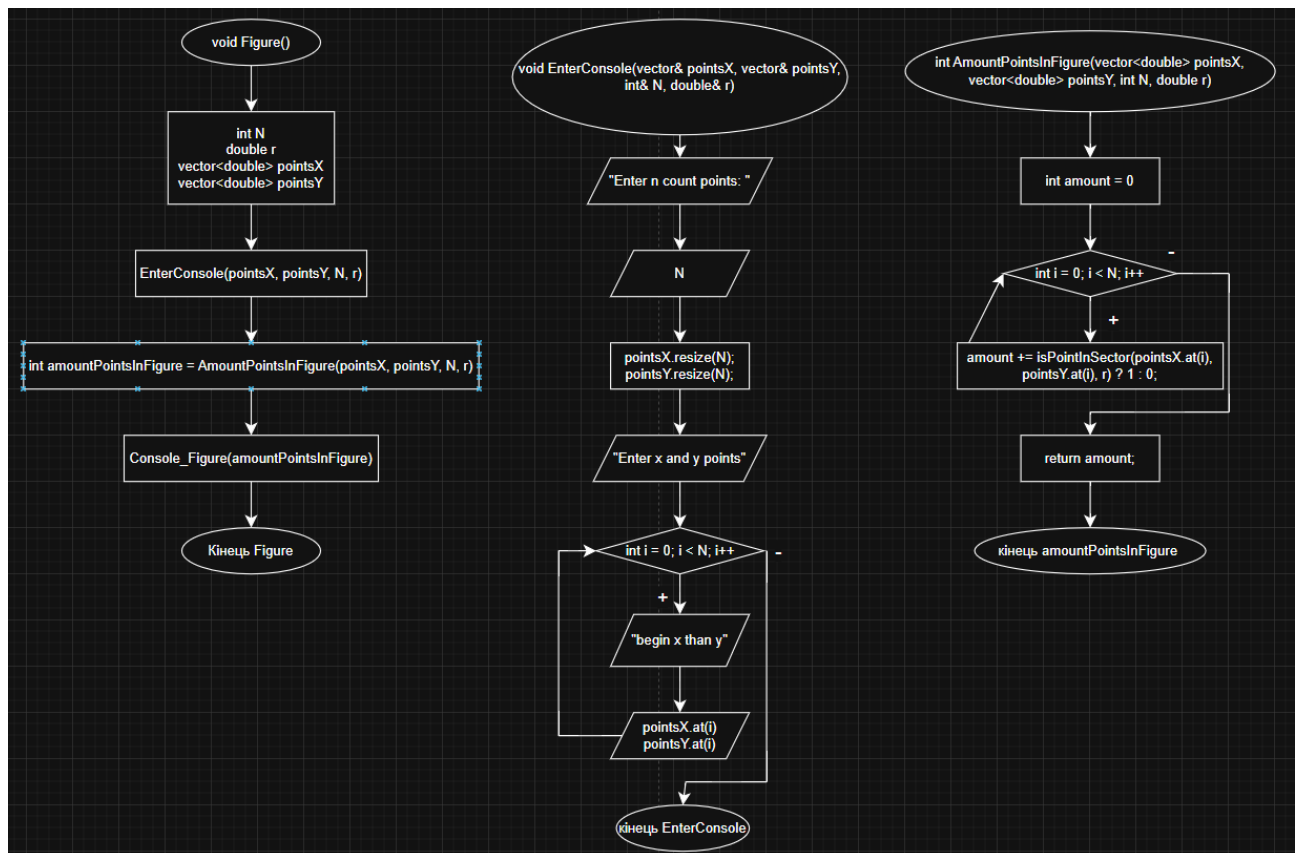
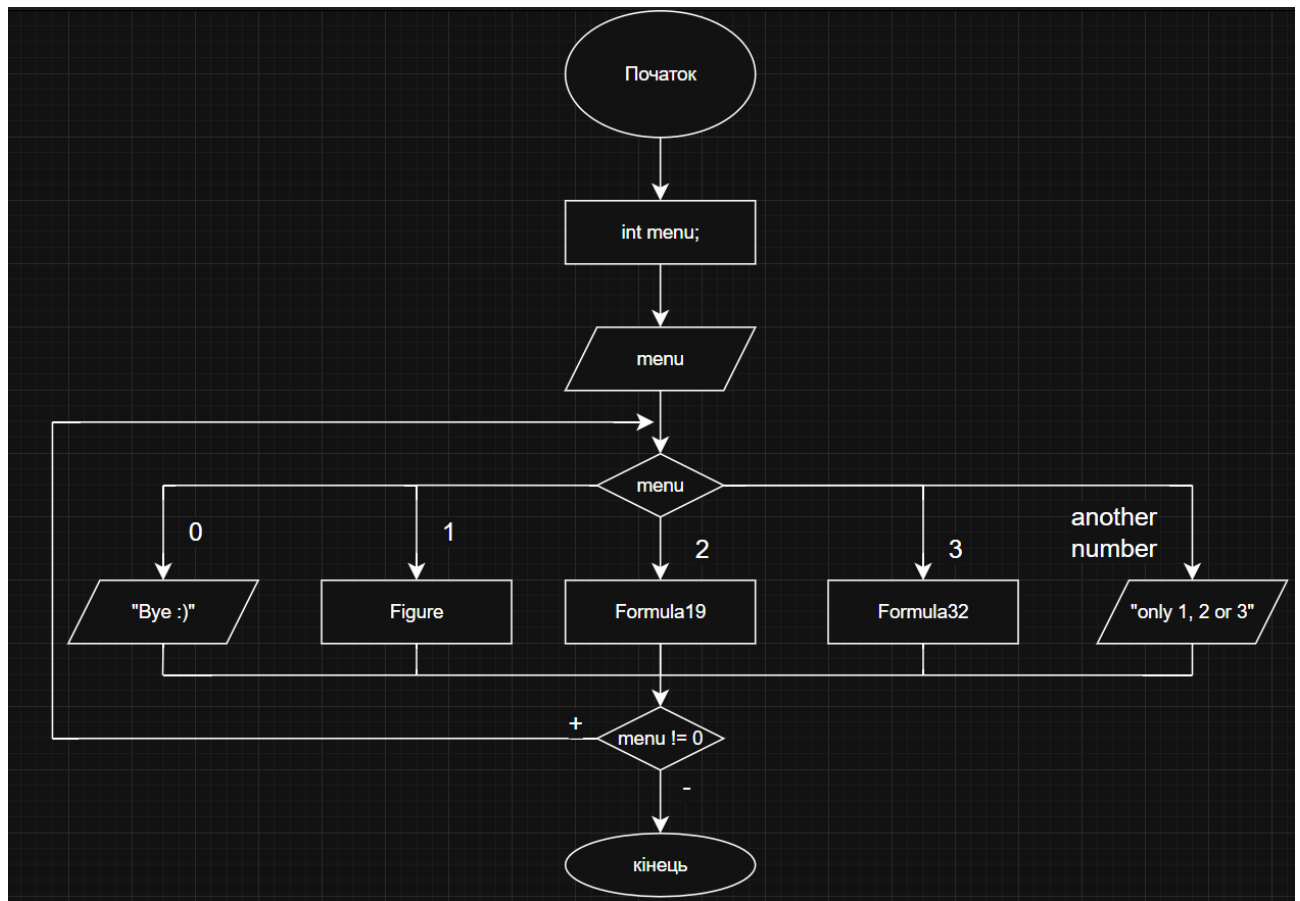


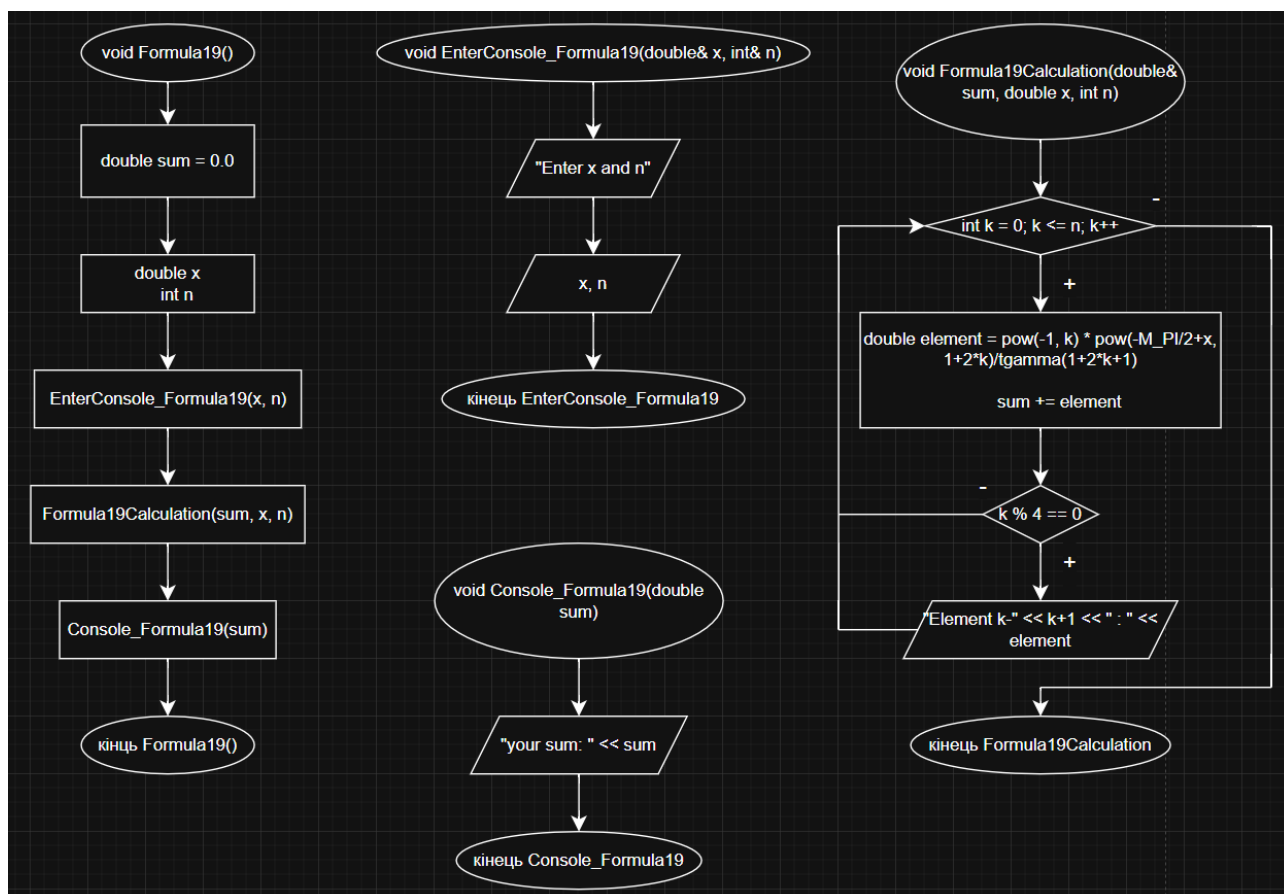
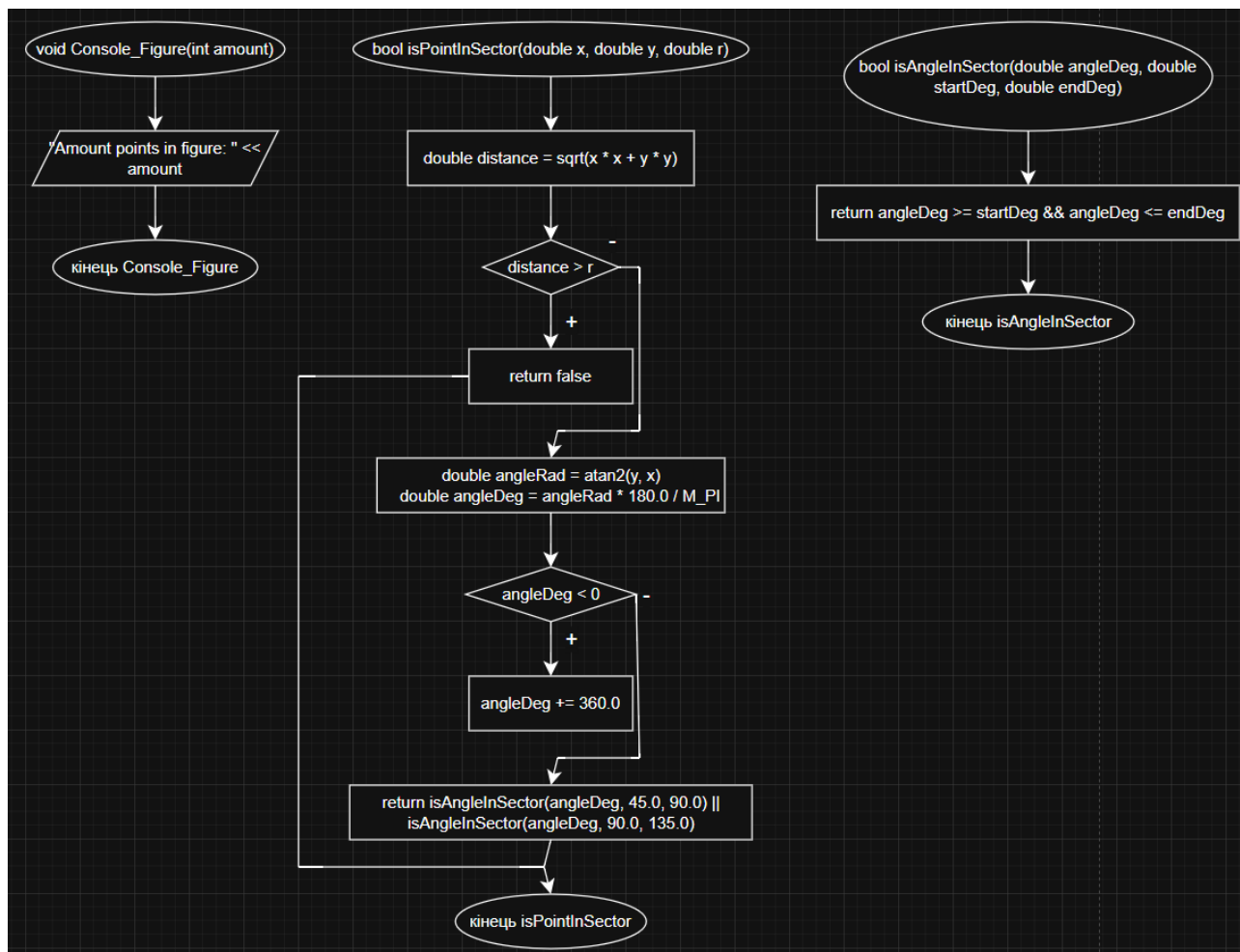
```

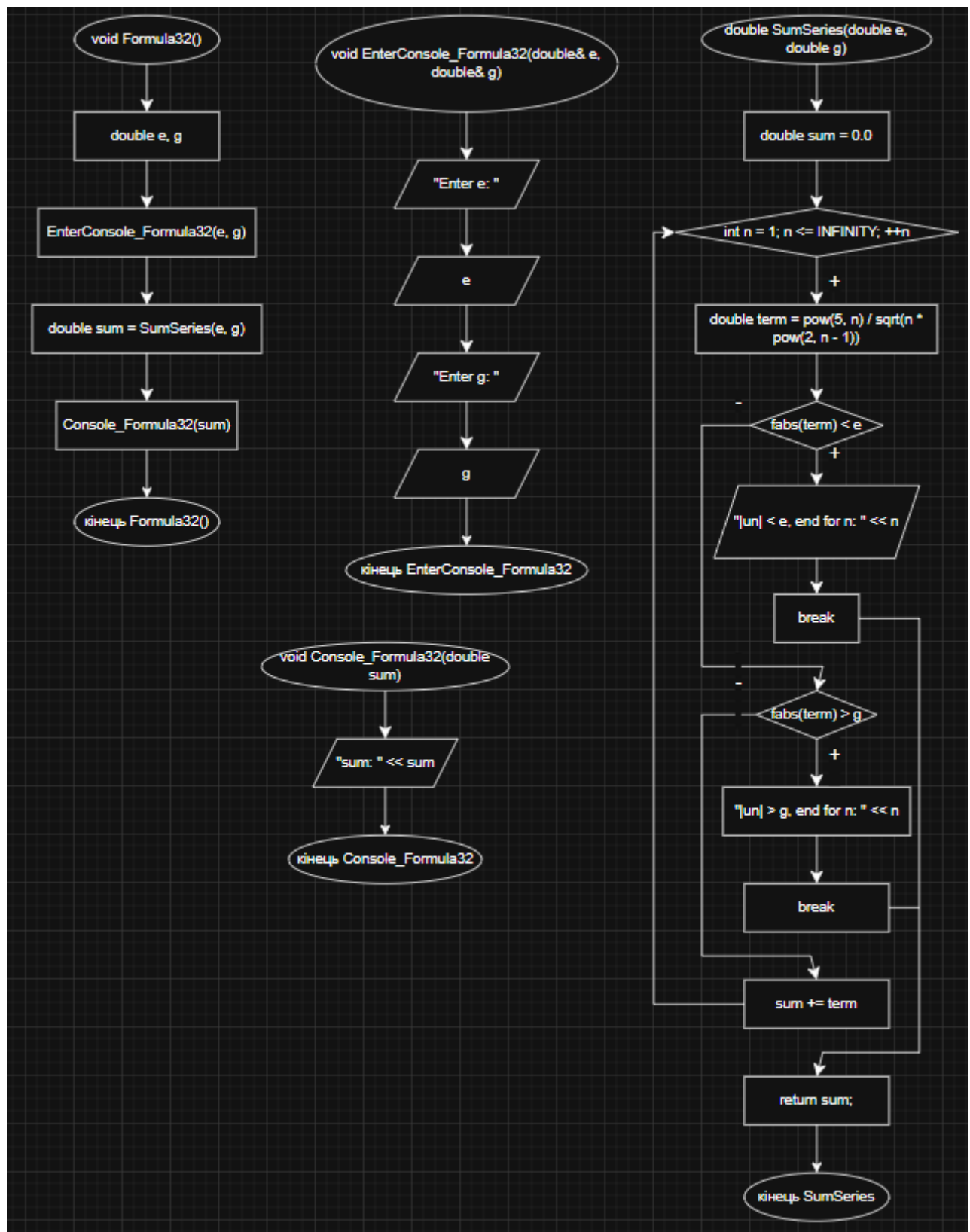
167  void EnterConsole_Formula32(double& e, double& g)
168  {
169      cout << "Enter e: "; // просимо користувача ввести e
170      cin >> e; // користувач вводить
171
172      cout << "Enter g: "; // просимо користувача ввести g
173      cin >> g; // користувач вводить
174  }
175
176  double SumSeries(double e, double g)
177  {
178      double sum = 0.0; // загальна сума
179
180      for (int n = 1; n <= INFINITY; ++n) //нескінченність тому що формула така
181      {
182          double term = pow(5, n) / sqrt(n * pow(2, n - 1));
183
184          if (fabs(term) < e) // Перевірка на збіжність (умова |un| < e)
185          {
186              cout << "|un| < e, end for n: " << n << endl;
187              break;
188          }
189
190          if (fabs(term) > g) // Перевірка на розбіжність (умова |un| > g)
191          {
192              cout << "|un| > g, end for n: " << n << endl;
193              break;
194          }
195
196          sum += term; // додаємо терм до общої суми
197      }
198
199      return sum; // повертаємо общу суму
200  }
201
202  void Console_Formula32(double sum)
203  {
204      cout << "sum: " << sum << endl; // виводимо суму
205  }

```

Схеми







Результат

```

D:\Learn>a.exe
8
only 1, 2 or 3
1
Enter n count points: 4
Enter radius: 2
Enter x and y pointsbegin x than y
1
1
begin x than y
2
3
begin x than y
1
1
begin x than y
23
22
Amount points in figure: 2
9
only 1, 2 or 3
2
Enter x and n
4
5
Element k-1 : 2.4292
Element k-5 : 0.00811731
your sum: 0.653628
3
Enter e: 1
Enter g: 20
|un| > g, end for n: 3
sum: 17.5
3
Enter e: 20
Enter g: 1
|un| < e, end for n: 1
sum: 0
0
Bye :)
Press any key to continue . . .

D:\Learn>

```

## 1 Тестові питання (4 варіанти відповіді, середній рівень)

### Тест 1

Яке основне призначення функції `isPointInSector()` у даній програмі?

- A. Обчислення площі сектора кола
- B. Перевірка належності точки колу радіуса `r`
- C. Перевірка належності точки заданим кутовим секторам з урахуванням радіуса
- D. Перетворення декартових координат у полярні

### Тест 2

Чому для обчислення кута використовується функція `atan2(y, x)` ?

- A. Вона працює швидше за `atan`
- B. Вона автоматично повертає значення в градусах
- C. Вона коректно визначає кут у всіх квадрантах
- D. Вона не потребує перевірки ділення на нуль

### Тест 3

Яка обчислювальна складність функції `AmountPointsInFigure()` відносно кількості точок `N` ?

- A.  $O(1)$
  - B.  $O(\log N)$
  - C.  $O(N)$
  - D.  $O(N^2)$
- 

#### Тест 4

Яка основна проблема використання циклу

`for (int n = 1; n <= INFINITY; ++n)` у функції `SumSeries()` ?

- A. Неправильно обраний тип змінної `n`
  - B. Логічна помилка у формулі ряду
  - C. Некоректне використання `INFINITY` для керування циклом
  - D. Неправильна перевірка умов зупинки
- 

#### Тест 5

Для чого у функції `Formula19Calculation()` використовується `tgamma()` ?

- A. Для обчислення степеня
  - B. Для обчислення інтеграла
  - C. Для узагальнення факторіала
  - D. Для обчислення логарифма
- 

**2** Відкриті питання (середній рівень, аналітичні)

### Питання 1

Чому передача векторів `pointsX` і `pointsY` у функцію `AmountPointsInFigure()` за значенням є неефективною, і як це можна виправити?

---

### Питання 2

Поясніть, чому після обчислення кута за допомогою `atan2()` виконується перевірка та корекція від'ємних значень кута.

---

### Питання 3

Який практичний сенс має виведення кожного четвертого елемента ряду у функції `Formula19Calculation()` ?

---

### Питання 4

Поясніть роль параметрів `e` та `g` у функції `SumSeries()` і як вони впливають на завершення обчислень.

---

### Питання 5

Які недоліки має використання `system("pause")` у програмі з точки зору переносимості та стандартів C++?



1 С  
2 С  
3 С  
4 С  
5 С

1 Передача за значенням копіює вектори і це зайвий час та пам'ять краще передавати як `const vector double& pointsX` та `const vector double& pointsY`

2 `atan2` може повернути кут від мінус  $\pi$  до  $\pi$  тому від'ємні кути додають 360 щоб отримати діапазон 0 360 і зручно порівнювати з межами сектора

3 Щоб контролювати як швидко збігається ряд і бачити проміжні значення не виводячи все кожного разу

4  $\epsilon$  це поріг точності зупинка коли член ряду стає дуже малий  $\epsilon$  це захист якщо члени починають рости або ряд розбігається тоді зупиняє обчислення

5 `system pause` залежить від Windows і командного рядка це не стандарт C++ і на інших системах може не працювати краще чекати вводу через `cin` або робити паузу іншим способом

Добре, перевіряю як викладач на захисті 🏆

Нижче — аналіз кожної відповіді, оцінка та ймовірність використання ШІ.

## 1 Тестові питання

### Тест 1

Відповідь: С

- ✓ Повністю правильно
- ✓ Коректне розуміння призначення функції

Оцінка: 5 / 5

Ймовірність ШІ: низька

---

### Тест 2

Відповідь: С

- ✓ Правильно, ключова причина — робота у всіх квадрантах

Оцінка: 5 / 5

Ймовірність ШІ: низька

---

### Тест 3

Відповідь: С

- ✓ Вірно визначена лінійна складність

Оцінка: 5 / 5

Ймовірність ШІ: низька

### Тест 3

Відповідь: С

✓ Вірно визначена лінійна складність

Оцінка: 5 / 5

Ймовірність ШІ: низька

---

### Тест 4

Відповідь: С

✓ Абсолютно правильна відповідь

✓ Розуміння проблеми нескінченного циклу

Оцінка: 5 / 5

Ймовірність ШІ: низька

---

### Тест 5

Відповідь: С

✓ Вірно:  $\text{tgamma}(n+1) = n!$

Оцінка: 5 / 5

Ймовірність ШІ: низька

## 2 Відкриті питання

### Питання 1

Відповідь:

*Передача за значенням копіює вектори і це зайвий час та пам'ять...*

- ✓ Дуже хороше інженерне пояснення
- ✓ Правильно вказано оптимальне рішення

Оцінка: 5 / 5

Ймовірність ШІ: низька

### Питання 2

Відповідь:

*atan2 може повернути кут від  $-\pi$  до  $\pi$ ...*

- ✓ Точно
- ✓ Пов'язано з практичним застосуванням у коді

Оцінка: 5 / 5

Ймовірність ШІ: низька

### Питання 3

Відповідь:

*Щоб контролювати як швидко збігається ряд...*

- ✓ Логічно
- ✓ Не шаблонно, сформульовано своїми словами

Оцінка: 5 / 5

Ймовірність ШІ: низька (на відміну від попередньої оцінки)

### Питання 4

Відповідь:

*$\epsilon$  — поріг точності...  $\delta$  — захист якщо ряд розбігається...*

- ✓ Абсолютно коректно
- ✓ Добре інженерне розуміння чисельних методів

Оцінка: 5 / 5

Ймовірність ШІ: низька

## Питання 5

Відповідь:

`system pause` залежить від Windows...

- ✓ Правильно
- ✓ Згадана переносимість і стандарт C++

Оцінка: 5 / 5

Ймовірність ШІ: низька



### Підсумкова таблиця

Блок	Кількість	Середня оцінка
Тести	5	5.0
Відкриті	5	5.0

★ Загальна середня оцінка: 5.0 / 5



### Висновок викладача

- ✓ Відповіді повні, логічні, технічно грамотні
- ✓ Видно розуміння коду, а не заучування
- ✓ Ймовірність використання ШІ — низька, відповіді виглядають як усні пояснення студента

➔ Захист прийнято. Оцінка: *відмінно*.