Made Abel Noelanza
22/494616/TK/54274

Senin, 27 Oktober 2025

User-Defined Function

1. True/False Question

a. To use a predefined function in a program, you need to know only the name of the function and how to use it. **(TRUE)**

b. A value-returning function returns only one value. **(TRUE)**

c. Parameters allow you to use different values each time the function is called. **(TRUE)**

d. When a return statement executes in a user-defined function, the function immediately exits. **(TRUE)**

e. A value-returning function returns only integer values. **(FALSE)**

f. A function that changes the value of a reference parameter also changes the value of the actual parameter. **(TRUE)**

g. A variable name cannot be passed to a value parameter. **(FALSE)**

h. If a C++ function does not use parameters, parentheses around the empty parameter list are still required. **(TRUE)**

i. In C++, the names of the corresponding formal and actual parameters must be the same. **(FALSE)**

j. Whenever the value of a reference parameter changes, the value of the actual parameter changes. **(TRUE)**

k. In C++, function definitions can be nested; that is, the definition of one function can be enclosed in the body of another function. **(FALSE)**

l. Using global variables in a program is a better programming style than using local variables, because extra variables can be avoided. **(FALSE)**

m. In a program, global constants are as dangerous as global variables. **(FALSE)**

n. The memoryfor a static variable remains allocated between function calls. **(FALSE)**

2. Chapter 6, Programming Exercise 8: Distance Between Two Points

Kode

```
#include <iostream>
#include <cmath>
using namespace std;

const double PI = 3.1416;

double getDistance(double xA, double yA, double xB, double
yB) {
    return sqrt(pow(xB - xA, 2) + pow(yB - yA, 2));
}

double getRadius(double xA, double yA, double xB, double
yB) {
    return getDistance(xA, yA, xB, yB);
}

double calcCircumference(double radius) {
    return 2 * PI * radius;
}

double calcArea(double radius) {
    return PI * pow(radius, 2);
}

int main() {
    double centerX, centerY, pointX, pointY;
    double radius, diameter, circumference, area;

    cout << "=== Circle Geometry Calculator ===\n";
    cout << "Enter center coordinates (x1 y1): ";
    cin >> centerX >> centerY;
```

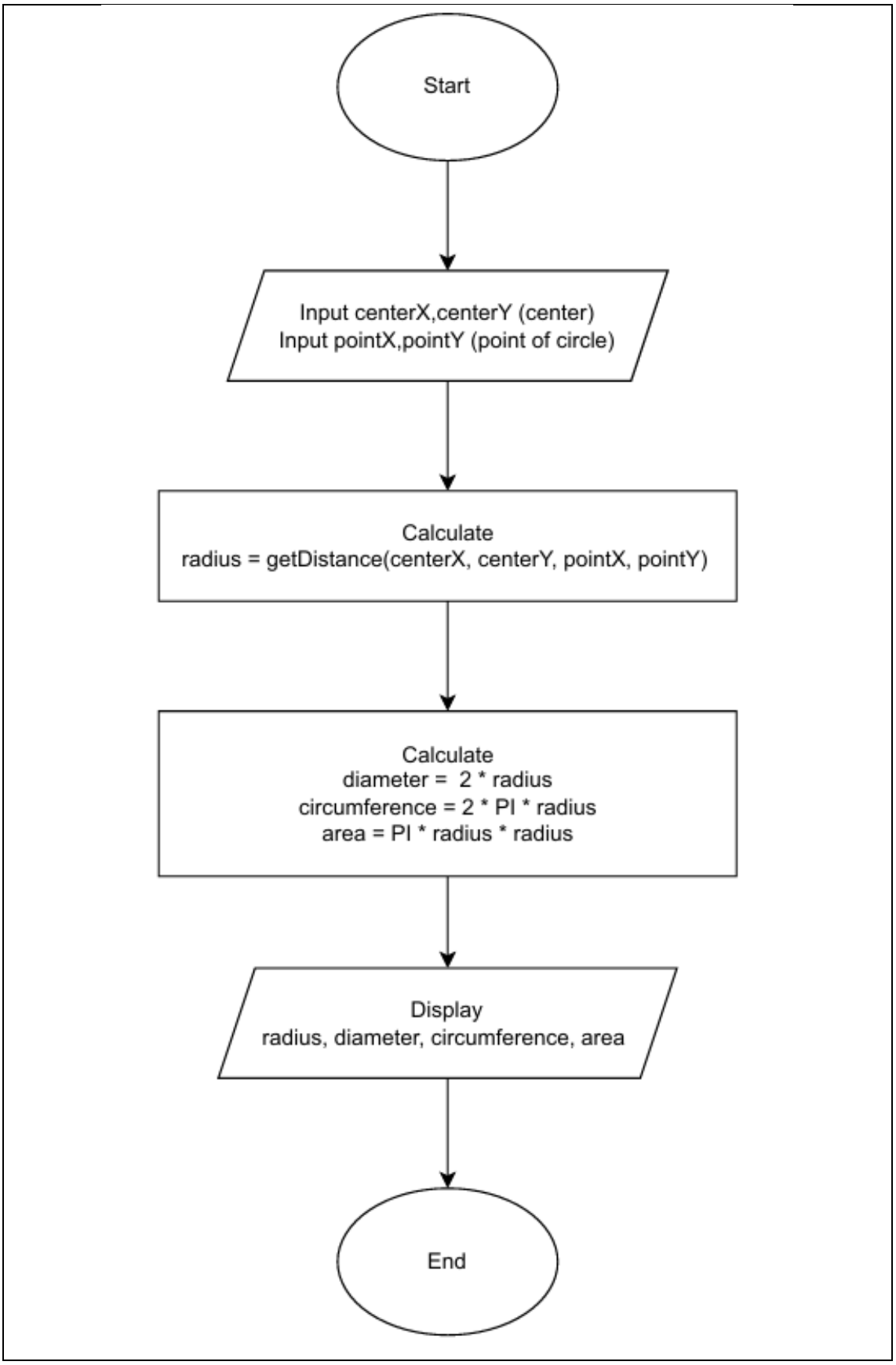
```
    cout << "Enter a point on the circle (x2 y2): ";
    cin >> pointX >> pointY;

    radius = getRadius(centerX, centerY, pointX, pointY);
    diameter = 2 * radius;
    circumference = calcCircumference(radius);
    area = calcArea(radius);

    cout << "\n=== Computation Results ===\n";
    cout << "Radius        : " << radius << endl;
    cout << "Diameter      : " << diameter << endl;
    cout << "Circumference : " << circumference << endl;
    cout << "Area          : " << area << endl;

    return 0;
}
```

Flowchart


Start
Input centerX,centerY (center)
Input pointX,pointY (point of circle)
Calculate
radius = getDistance(centerX, centerY, pointX, pointY)
Calculate
diameter = 2 * radius
circumference = 2 * PI * radius
area = PI * radius * radius
Display
radius, diameter, circumference, area
End

Alur Program

1) Deklarasi library dan konstanta

`#include <iostream>` → digunakan untuk fungsi input/output seperti cin dan cout.

`#include <cmath>` → digunakan untuk fungsi matematika seperti sqrt() dan pow().

`const double PI = 3.1416;` → mendefinisikan nilai π (pi) yang akan digunakan untuk perhitungan keliling dan luas lingkaran

2) Definisi fungsi

a. getDistance()

`double getDistance(double xA, double yA, double xB, double yB)`

Fungsi ini menghitung jarak antara dua titik (titik pusat dan titik pada keliling) dengan rumus:

$$\text{Jarak} = \sqrt{(x_B - x_A)^2 + (y_B - y_A)^2}$$

Fungsi ini mengembalikan hasil jarak dalam bentuk bilangan double.

b. getRadius()

`double getRadius(double xA, double yA, double xB, double yB)`

Fungsi ini memanggil fungsi getDistance() untuk mencari jarak antara kedua titik tersebut. Karena jarak dari pusat ke titik di keliling sama dengan jari-jari (radius) lingkaran, maka hasil jarak tadi dianggap sebagai radius.

c. calcCircumference()

`double calcCircumference(double radius)`

Fungsi ini menghitung keliling lingkaran dengan rumus:

$$C = 2 \times \pi \times r$$

Nilainya dikembalikan (return) ke pemanggil fungsi.

d. calcArea()

`double calcArea(double radius)`

Fungsi ini menghitung luas lingkaran menggunakan rumus:

$$A = \pi \times r^2$$

dan mengembalikannya ke pemanggil fungsi.

3) Fungsi utama (main())

```
int main() {
    double centerX, centerY, pointX, pointY;
    double radius, diameter, circumference, area;
```

Di sini program menyiapkan variabel untuk menyimpan:

- Koordinat pusat dan titik lingkaran
- Hasil perhitungan: jari-jari, diameter, keliling, dan luas

4) Input dari pengguna

```
cout << "Enter center coordinates (x1 y1): ";
cin >> centerX >> centerY;
cout << "Enter a point on the circle (x2 y2): ";
cin >> pointX >> pointY;
```

Program meminta pengguna memasukkan dua pasang koordinat:

- (x1, y1) untuk pusat
- (x2, y2) untuk titik pada keliling

Semua nilai dibaca menggunakan cin dan disimpan ke variabel yang sesuai.

5) Proses perhitungan

Setelah semua input diterima, program melakukan perhitungan berurutan:

a. `radius = getRadius(centerX, centerY, pointX, pointY);`

Memanggil fungsi getRadius() untuk menghitung jari-jari lingkaran berdasarkan dua titik.

b. `diameter = 2 * radius;`

Menghitung diameter dengan mengalikan jari-jari × 2.

c. `circumference = calcCircumference(radius);`

Memanggil fungsi calcCircumference() untuk menghitung keliling lingkaran.

d. `area = calcArea(radius);`

Memanggil fungsi calcArea() untuk menghitung luas lingkaran.

6) Menampilkan hasil

```
cout << "\n=== Computation Results ===\n";
cout << "Radius        : " << radius << endl;
```

```
cout << "Diameter      : " << diameter << endl;
cout << "Circumference : " << circumference << endl;
cout << "Area          : " << area << endl;
```

Setelah semua nilai didapat, program menampilkan hasil ke layar:

- Nilai radius
- Nilai diameter
- Nilai keliling
- Nilai luas

Tiap hasil ditampilkan dengan label agar mudah dibaca.

Output

```
=== Circle Geometry Calculator ===
Enter center coordinates (x1 y1): 2 3
Enter a point on the circle (x2 y2): 5 7

=== Computation Results ===
Radius        : 5
Diameter      : 10
Circumference : 31.416
Area          : 78.54
```

3. Formal Parameter, Actual Parameter, & Function Signature

a. Formal Parameter (Parameter Formal)

Formal parameter adalah variabel sementara yang tertulis di dalam definisi fungsi. Mereka berfungsi untuk menerima nilai dari argumen (nilai sebenarnya) ketika fungsi dipanggil.

Contoh:

- `double getDistance(double xA, double yA, double xB, double yB)` → xA, yA, xB, yB adalah parameter formal. Mereka belum memiliki nilai sampai fungsi ini dipanggil oleh main().
- `double calcCircumference(double radius);` // radius → formal parameter
- `double calcArea(double radius);` // radius → formal parameter

b. Actual Parameter (Parameter Aktual)

Actual parameter adalah nilai sesungguhnya yang dikirim ke fungsi ketika fungsi dipanggil. Nilai ini bisa berupa variabel atau konstanta.

Contoh :

- `radius = getRadius(centerX, centerY, pointX, pointY);` → centerX, centerY, pointX, pointY adalah actual parameters karena mereka adalah nilai nyata yang dikirim ke parameter formal (xA, yA, xB, yB) di fungsi getRadius().
- `circumference = calcCircumference(radius);` // radius = actual parameter
- `area = calcArea(radius);` // radius = actual parameter

c. Function Signature

Function signature adalah *bentuk atau struktur dasar* dari fungsi yang menunjukkan:

- Nama fungsi
- Tipe data dari parameter-parameternya (termasuk jumlah dan urutannya) Tanda tangan fungsi tidak mencakup tipe data pengembalian (return type).

Fungsi	Function Signature
`getDistance`	`getDistance(double, double, double, double)`
`getRadius`	`getRadius(double, double, double, double)`
`calcCircumference`	`calcCircumference(double)`
`calcArea`	`calcArea(double)`

Misalnya, `double getDistance(double xA, double yA, double xB, double yB)` → Signature-nya hanyalah getDistance(double, double, double, double) karena double di depan (return type) tidak dihitung dalam signature.

Lampiran link *source code*:
https://github.com/MadeAbelNoelanza/Tugas-Pemrograman-Dasar-User-Defined-Function.git