Struktur Program C++

Pertemuan 3

Outline

- Organizing the C++ Program
  - Declaring and Defining Functions
  - Local Variables
  - Global Variables
  - Returning Values from Functions
  - Default Function Parameters
  - Controlling the Flow of a Program
  - **while** Loops
  - Breaking Out of Loops
  - Continuing to the Next Loop
  - **do**-**while** Loops
  - **for** Loops
  - Nested Loops
  - **switch** Statements

Deklarasi & Definisi Fungsi

- Deklarasi fungsi memberitahu compiler nama fungsi, tipe data fungsi, dan tipe parameter yang diterima oleh fungsi. Deklarasi Fungsi, yang juga disebut prototipe, tidak berisi kode. Berikut adalah deklarasi untuk fungsi yang menentukan wilayah persegi panjang dengan menggunakan panjang dan lebar parameter:

```
int findArea(int length, int width);
```

- Keterangan
  - Return type : int
  - Function name : findArea
  - Parameters : int length, int width

Variabel Local & Global

- Sebuah variabel dibuat dalam suatu fungsi disebut variabel lokal karena hanya bekerja dalam fungsi itu sendiri. Ketika fungsi mengembalikan, semua variabel lokal tidak lagi tersedia untuk digunakan dalam program ini.
- Variabel dapat didefinisikan di luar semua fungsi dalam program C ++ termasuk main (). Ini disebut variabel global karena variabel ini dapat digunakan di semua fungsi dalam program.

Parameter & Return Value

- Fungsi menerima informasi dalam bentuk parameter fungsi. Boleh tidak ada parameter, atau bisa lebih dari satu parameter selama mereka dipisahkan dengan koma.

```
int x = 4, y = 13;
swap(x, y);
void swap(int x, int y) {
        int temp = x;
        int x = y;
        int y = temp;
}
```

- Fungsi dapat mengembalikan nilai atau tidak. Untuk mengembalikan nilai dari fungsi, kata kunci adalah “return” diikuti oleh nilai, variabel, atau ekspresi.

```
return 5;
return (x > 5);
return (convert(fahrenheit));
```

Fungsi Overloading dan Inline

- Dalam C ++, lebih dari satu fungsi dapat memiliki nama yang sama selama ada perbedaan dalam argumen mereka (tipe data, jumlah argumen), hal ini disebut overloading.

```
int store(int, int);
int store(long, long);
int store(long);
```

- Jika fungsi dideklarasikan dengan kata kunci “inline”, compiler tidak akan membuat fungsi secara nyata. Sebaliknya, itu adalah salinan kode dari fungsi inline yang langsung menuju ke tempat di mana fungsi dipanggil.

```
inline int double(int);
```

1. Declaring and Defining Functions

Area.cpp

```
1: #include <iostream>
2:
3: int findArea(int length, int width); // function prototype
4:
5: int main()
6: {
7:        int length;
8:        int width;
9:        int area;
10:
11:       std::cout << "\nHow wide is your yard? ";
12:       std::cin >> width;
13:       std::cout << "\nHow long is your yard? ";
14:       std::cin >> length;
15:
16:       area = findArea(length, width);
17:
18:       std::cout << "\nYour yard is ";
19:       std::cout << area;
20:       std::cout << " square feet\n\n";
21:       return 0;
22: }
23:
24: // function definition
25: int findArea(int l, int w)
26: {
27:       return l * w;
28: }
```

2. Local Variables

Temperature.cpp

```
1: #include <iostream>
2:
3: float convert(float);
4:
5: int main()
6: {
7:        float fahrenheit;
8:        float celsius;
9:
10:       std::cout << "Please enter the temperature in Fahrenheit: ";
11:       std::cin >> fahrenheit;
12:       celsius = convert(fahrenheit);
13:       std::cout << "\nHere's the temperature in Celsius: ";
14:       std::cout << celsius << "\n";
15:       return 0;
16: }
17:
18: // function to convert Fahrenheit to Celsius
19: float convert(float fahrenheit)
20: {
21:       float celsius;
22:       celsius = ((fahrenheit - 32) * 5) / 9;
23:       return celsius;
24: }
```

3. Global Variables

Global.cpp

```
1: #include <iostream>
2:
3: void convert();
4:
5: float fahrenheit;
6: float celsius;
7:
8: int main()
9: {
10:
11:       std::cout << "Please enter the temperature in Fahrenheit: ";
12:       std::cin >> fahrenheit;;
13:       convert();
14:       std::cout << "\nHere's the temperature in Celsius: ";
15:       std::cout << celsius << "\n";
16:       return 0;
17: }
18:
19: // function to convert Fahrenheit to Celsius
20: void convert()
21: {
22:       celsius = ((fahrenheit - 32) * 5) / 9;
23: }
```

4. Returning Values from Functions

LeapYear.cpp

```
1: #include <iostream>
2:
3: bool isLeapYear(int year);
4:
5: int main()
6: {
7:         int input;
8:         std::cout << "Enter a year: ";
9:         std::cin >> input;
10:        if (isLeapYear(input))
11:        {
12:                  std::cout << input << " is a leap year\n";
13:        }
14:        else
15:        {
16:                  std::cout << input << " is not a leap year\n";
17:        }
18:        return 0;
19: }
20:
```

...

```
21: bool isLeapYear(int year)
22: {
23:       if (year % 4 == 0)
24:       {
25:                 if (year % 100 == 0)
26:                 {
27:                           if (year % 400 == 0)
28:                           {
29:                                     return true;
30:                           }
31:                           else
32:                           {
33:                                     return false;
34:                           }
35:                 }
36:                 else
37:                 {
38:                           return true;
39:                 }
40:       }
41:       else
42:       {
43:                 return false;
44:       }
45: }
```

5. Default Function Parameters

AreaCube.cpp

```
1: #include <iostream>
2:
3: int findArea(int length, int width = 20, int height = 12);
4:
5: int main()
6: {
7:         int length = 100;
8:         int width = 50;
9:         int height = 2;
10:        int area;
11:
12:        area = findArea(length, width, height);
13:        std::cout << "First area: " << area << "\n\n";
14:
15:        area = findArea(length, width);
16:        std::cout << "Second area: " << area << "\n\n";
17:
18:        area = findArea(length);
19:        std::cout << "Third area: " << area << "\n\n";
20:        return 0;
21: }
22:
23: int findArea(int length, int width, int height)
24: {
25:        return (length * width * height);
26: }
```

6. **while** Loops

Thirteens.cpp

```
1: #include <iostream>
2:
3: int main()
4: {
5: int counter = 0;
6:
7: while (counter < 500)
8: {
9:        counter++;
10:       if (counter % 13 == 0)
11:       {
12:                 std::cout << counter << " ";
13:       }
14: }
15:
16:       std::cout << "\n";
17:       return 0;
18: }
```

7. Breaking Out of Loops

Fourteens.cpp

```
1: #include <iostream>
2:
3: int main()
4: {
5:         int counter = 0;
6:         int multiples = 0;
7:
8: while (true)
9: {
10:        counter++;
11:        if (counter % 14 == 0)
12:        {
13:                  std::cout << counter << " ";
14:                  multiples++;
15:        }
16:        if (multiples > 19)
17:        {
18:                  break;
19:        }
20: }
21:
22:        std::cout << "\n";
23:        return 0;
24: }
```

8. Continuing to the Next Loop

Fifteens.cpp

```
1: #include <iostream>
2:
3: int main()
4: {
5:         int counter = 0;
6:         int multiples = 0;
7:
8: while (multiples < 19)
9: {
10:        counter++;
11:        if (counter % 15 != 0)
12:        {
13:        continue;
14:        }
15:                  std::cout << counter << " ";
16:                  multiples++;
17:        }
18:
19:        std::cout << "\n";
20:        return 0;
21: }
```

9. **do-while** Loops

Badger.cpp

```
1: #include <iostream>
2:
3: int main()
4: {
5:       int badger;
6:       std::cout << "How many badgers? ";
7:       std::cin >> badger;
8:
9:  do
10: {
11:      std::cout << "Badger ";
12:      badger--;
13: } while (badger > 0);
14:
15:      std::cout << "\n";
16:      return 0;
17: }
```

10. **for** Loops

MultTable.cpp

```
1: #include <iostream>
2:
3: int main()
4: {
5:         int number;
6:         std::cout << "Enter a number: ";
7:         std::cin >> number;
8:
9:         std::cout << "\nFirst 10 Multiples of " << number << "\n";
10:
11:        for (int counter = 1; counter < 11; counter++)
12:        {
13:                  std::cout << number * counter << " ";
14:        }
15:        std::cout << "\n";
16:
17:        return 0;
18: }
```

11. Nested Loops

BoxMaker.cpp

```
1: #include <iostream>
2:
3: int main()
4: {
5:         int rows, columns;
6:         char character;
7:
8:         std::cout << "How many rows? ";
9:         std::cin >> rows;
10:        std::cout << "How many columns? ";
11:        std::cin >> columns;
12:        std::cout << "What character to display? ";
13:        std::cin >> character;
14:
15:        std::cout << "\n";
16:        for (int i = 0; i < rows; i++)
17:        {
18:                  for (int j = 0; j < columns; j++)
19:                  {
20:                           std::cout << character;
21:                  }
22:                  std::cout << "\n";
23:        }
24:        return 0;
25: }
```

12. **switch** Statements

BadTeacher.cpp

```
1: #include <iostream>
2:
3: int main()
4: {
5:         char grade;
6:         std::cout << "Enter your letter grade (ABCDF): ";
7:         std::cin >> grade;
8:         switch (grade)
9:         {
10:                 case 'A':
11:                 std::cout << "Finally!\n";
12:                 break;
13:                 case 'B':
14:                 std::cout << "You can do better!\n";
15:                 break;
16:                 case 'C':
17:                 std::cout << "I'm disappointed in you!\n";
18:                 break;
19:                 case 'D':
20:                 std::cout << "You're not smart!\n";
21:                 break;
22:                 case 'F':
23:                 std::cout << "Get out of my sight!\n";
24:                 break;
25:                 default:
26:                 std::cout << "That's not even a grade!\n";
27:                 break;
28:         }
29:         return 0;
30: }
```

Tugas

- Buatlah sebuah program yang dapat mengkonversi suhu dari Celcius ke Fahrenheit, dengan menggunakan formula : kalikan temperatur celcius dengan 9, kemudian dibagi hasilnya dengan 5, kemudian ditambahkan dengan 32.
- Buatlah sebuah program yang dapat menghitung rata-rata 2 buah bilangan integer, 2 buah long integer atau 2 buah floating-point dengan menggunakan overloaded function yang bernama average()
- Buatlah program yang dapat menampilkan 100 bilangan yang merupakan kelipatan dari 16.
- Modifikasi program BadTeacher.cpp agar dapat menangani grade E dan G melewati H dan menampilkan respon secara kasar disesuaikan dengan masukan yang diberikan.