THE FUNCTION OF WORKING WITH TEMPLATES IS HOW TO FIX TEMPLATES AND SOLVE PROBLEMS RELATED TO THEM

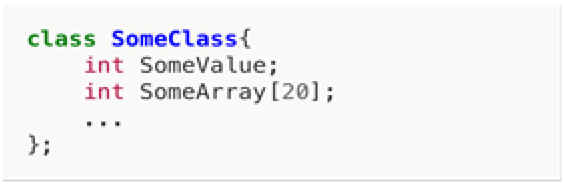
SH.M.Samandarova
(Named after Muhammad Al-Xorazmiy TATU karshi branch teacher of the software engineering department)
Ahmedova Nafisa
(Student of TT 11-23, Faculty of Telecommunication Technologies, Karshi Branch, Tashkent University of Information Technologies named after Muhammad al-Khorazmi)

Annotation: *This article is intended to show how to solve classes in C++ using templates.*
Key words: *Templates, C++ language, class, SomeValue and SomeArray.*

Templates are a C++ language tool designed for coding generalized algorithms without reference to certain parameters (for example, data types, buffer sizes, default values).

In C++ it is possible to create function and class templates.

Templates allow you to create parameterized classes and functions. The parameter can be any type or a value of one of the valid types (integer, enum, pointer to any object with a globally accessible name, reference). For example, we need some kind of class:

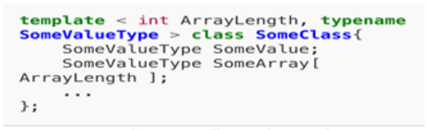


```
class SomeClass{
    int SomeValue;
    int SomeArray[20];
    ...
};
```

1-picture Template to announce.

For one specific purpose we can use this class. But, suddenly, the goal has changed a little, and another class is needed. Now you need 30 elements of the SomeArray array and the real type SomeValue of the SomeArray elements.

Then we can abstract away from concrete types and use templates with parameters. Syntax: at the beginning, before declaring the class, we declare a template, that is, template, and indicate the parameters in angle brackets. In our example:



```
template < int ArrayLength, typename
SomeValueType > class SomeClass{
    SomeValueType SomeValue;
    SomeValueType SomeArray[
ArrayLength ];
    ...
};
```

2-picture Example template.

Then for the first case (with an integer **SomeValue** and **SomeArray** of 20 elements) we write:

```
SomeClass < 20, int > SomeVariable;
```

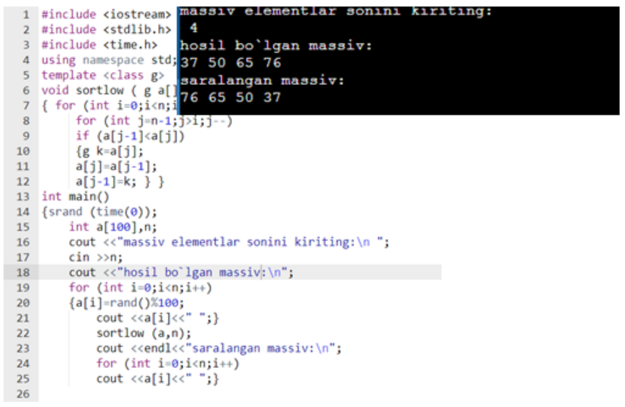
for the second:

```
SomeClass < 30, double >
SomeVariable2;
```

Although templates provide a short form for writing a section of source code, their use does not shorten the executable code, since the compiler creates a separate instance of the function or class for each set of parameters. As a consequence, the ability to share compiled code within shared libraries disappears.

Template description syntax editing

A function template begins with a template keyword followed by a list of parameters in angle brackets. The following is the function declaration:



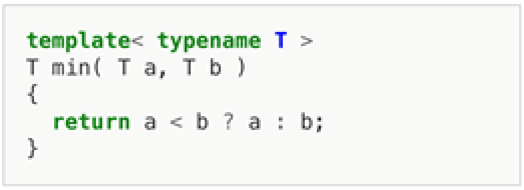
```
#include <iostream>
#include <stdlib.h>
#include <time.h>
using namespace std;
template <class g>
void sortlow ( g a[
{ for (int i=0;i<n;i
      for (int j=n-1;j>i;j--)
      if (a[j-1]<a[j])
      {g k=a[j];
      a[j]=a[j-1];
      a[j-1]=k; } }
int main()
{srand (time(0));
     int a[100],n;
     cout <<"massiv elementlar sonini kiriting:\n ";
     cin >>n;
     cout <<"hosil bo`lgan massiv:\n";
     for (int i=0;i<n;i++)
     {a[i]=rand()%100;
         cout <<a[i]<<" ";}
         sortlow (a,n);
         cout <<endl<<"saralangan massiv:\n";
         for (int i=0;i<n;i++)
         cout <<a[i]<<" ";}

```

```
massiv elementlar sonini kiriting:
 4
hosil bo`lgan massiv:
37 50 65 76
saralangan massiv:
76 65 50 37
```

3-picture Example template.

The simplest example is determining the minimum of two quantities. If a is less than b, then return a, otherwise return b Without templates, the programmer must write separate functions for each data type used. Although many programming languages define a built-in minimum function for elementary types (such as integers and real numbers), such a function may also be needed for complex ones (such as “time” or “string”) and very complex ones (“player” in an online game). objects. This is what the minimum function template looks like:



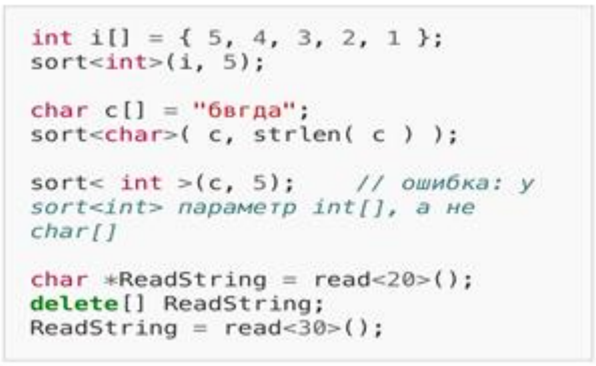
```
template< typename T >
T min( T a, T b )
{
  return a < b ? a : b;
}
```

4-picture Example template.

To call this function you can simply use its name:

```
min( 1, 2 );
min( 'a', 'b' );
min( string( "abc" ), string( "cde"
) );
```

Generally speaking, to call a template function, you must specify values for all template parameters. To do this, after the template name, indicate a list ofvalues in angle brackets:



```
int i[] = { 5, 4, 3, 2, 1 };
sort<int>(i, 5);

char c[] = "бвгда";
sort<char>( c, strlen( c ) );

sort< int >(c, 5);     // ошибка: у
sort<int> параметр int[], а не
char[]

char *ReadString = read<20>();
delete[] ReadString;
ReadString = read<30>();
```

For each set of parameters, the compiler generates a new instance of the function. The process of creating a new instance is called template instantiation.

In the example above, the compiler created two specializations of the sort function template (for types char and int) and two specializations of the read template (for BufferSize values 20 and 30). The latter is most likely wasteful, since for each possible value of the parameter the compiler will create new and new instances of functions that will differ only in one constant.

REFERENCES:

1. David Vandevoord, Nikolai M. Josattis. C++ Templates: The Complete Guide. - M.: Williams, 2003. - P. 544. - ISBN 0-201-73484-2.
2. Podbelsky V.V. 6.9. Function templates //Chapter 6. Functions, pointers, references // C++ Language / Rec. Dadaev Yu. G. - 4. - M.: Finance and Statistics, 2003. - P. 230-236. — 560 s. — ISBN 5-279-02204-7, UDC 004.438Ci(075.8) BBK 32.973.26-018 1я173.

3. Samandarova, S. M., & Jabborov, E. E. (2023, November). The Priority Of Developing The Intellectual Potential Of Students In The Conditions Of Informatization Of Education. In international scientific research conference (Vol. 2, No. 17, pp. 85-87).
4. Mirjamilovna, S. S., Farxodovna, A. N., & Farxodovna, M. D. (2024). application of bubble sorting algorithm in swot analysis. formation of psychology and pedagogy as interdisciplinary sciences, 3(26), 192-196.
5. Farxodovna, A. N., Mirjamilovna, S. S., & Farxodovna, M. D. (2023). Yechimni avto-testlovchi tizimlardagi asosiy terminlar. Reytinglash jarayoning matematik modeli. O'zbekistonda fanlararo innovatsiyalar va ilmiy tadqiqotlar jurnali, 2(24), 151-155.