Дәріс 12. Шаблондар (Templates)

1. Шаблон дегеніміз не?

Шаблон (template) – бұл **деректер типіне тәуелсіз** код жазуға мүмкіндік беретін механизм.
Ол арқылы бір ғана функция немесе класс жазып, оны әртүрлі типтермен қайта-қайта қолдануға болады.

Шаблондар – **жалпылама бағдарламалаудың (generic programming)** негізгі құралы.

2. Функция-шаблондар (Function Templates)

Функция-шаблон – бір функцияны бірнеше типтермен қолдануға мүмкіндік береді. Мысалы, int, double, float сияқты әртүрлі типтер үшін жеке-жеке функция жазудың орнына, бір ғана шаблон жазамыз.

Синтаксис:

```
template <typename T>
T функцияАты(T a, T b) {
   // ...
}
```

Мысал: Екі мәннің максимумын табу

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

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

int main() {
   cout << maxVal(5, 9) << endl;       // int
   cout << maxVal(3.2, 7.5) << endl;    // double
   cout << maxVal('a', 'z') << endl;    // char
   return 0;
}
```

Бір ғана функция – үш түрлі типке қолданылды.

3. Бірнеше параметрлі шаблондар

Шаблон бірнеше тип қабылдай алады:

```
template <typename T1, typename T2>
```

```
void show(T1 a, T2 b) {
   cout << a << " және " << b << endl;
}

int main() {
   show(10, 3.14);
   show("Hello", 42);
}
```

4. Класс-шаблондар (Class Templates)

Класс-шаблон – әртүрлі типтерге арналған кластарды бір рет жазып, қайта пайдалануға мүмкіндік береді.

Синтаксис:

```
template <typename T>
class ClassName {
private:
   T data;
public:
   ClassName(T d) : data(d) {}
   void show() { cout << "Мән: " << data << endl; }
};
```

Мысал: Жалпы контейнер класы

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

template <typename T>
class Box {
private:
   T value;
public:
   Box(T v) : value(v) {}
   void show() {
      cout << "Мән: " << value << endl;
   }
};

int main() {
   Box<int> b1(100);
   Box<double> b2(3.14);
   Box<string> b3("Сәлем");
```

```
    b1.show();
    b2.show();
    b3.show();

    return 0;
}
```

Бір ғана класс – үш түрлі типпен жұмыс істейді.

5. Класс-шаблон әдістерін сыртта анықтау

Шаблон әдістерін класс сыртында анықтау үшін синтаксис сәл өзгереді:

```
template <typename T>
class Box {
private:
    T value;
public:
    Box(T v);
    void show();
};

template <typename T>
Box<T>::Box(T v) : value(v) {}

template <typename T>
void Box<T>::show() {
    cout << "Мән: " << value << endl;
}
```

6. Жалпылама бағдарламалау (Generic Programming)

Жалпылама бағдарламалау – алгоритмдер мен деректер құрылымдарын нақты типтерден тәуелсіз етіп жазу әдісі.
C++ шаблондар арқылы мұны жүзеге асырады.

Артықшылықтары:

- Код қайталанбайды
- Түрге тәуелсіздіктен **икемділік**
- Компиляция кезінде тип қауіпсіздігі қамтамасыз етіледі
- Жоғары өнімділік

7. Шаблондар мен полиморфизм айырмашылығы

Ерекшелік	Шаблондар	Полиморфизм
Орындау уақыты	Компиляция кезінде	Орындау кезінде
Тәсіл	Жалпылама бағдарламалау	Виртуалды функциялар арқылы
Икемділігі	Типке тәуелсіз код	Әртүрлі кластарда ортақ интерфейс

8. Арнайылау (Template Specialization)

Кейде белгілі бір тип үшін шаблонды ерекше етіп жазу қажет болады.

Мысал:

```
template <typename T>
void show(T a) {
   cout << "Жалпы: " << a << endl;
}

// Арнайылау string үшін
template <>
void show<string>(string s) {
   cout << "Мәтін: " << s << endl;
}

int main() {
   show(42);         // Жалпы нұсқа
   show(3.14);       // Жалпы нұсқа
   show<string>("Сәлем");  // Арнайы нұсқа
}
```

Қысқаша қорытынды

Түсінік	Мағынасы
Шаблон (template)	Түрге тәуелсіз функция/класс үлгісі
Функция-шаблон	Бір функцияны бірнеше типке қолдануға мүмкіндік береді
Класс-шаблон	Бір класс үлгісімен әртүрлі типтерге объектілер жасауға мүмкіндік береді
Арнайылау	Белгілі бір тип үшін шаблонды өзгеше анықтау
Жалпылама бағдарламалау	Түрге тәуелсіз, қайта пайдалануға болатын код жазу әдісі

Шаблондардың артықшылықтары:

- Код қысқарады және қайта қолданылады
- Әртүрлі деректер типтерімен жұмыс істейді
- Компиляция кезінде тип тексеріледі
- Жоғары өнімділік береді

Есте сақтау:

- template <typename T> – шаблон анықтауда қолданылады.
- Шаблондар тек компиляция кезінде кеңейтіледі.
- Класс және функция шаблондарын бірге қолдануға болады