«ԾՐԱԳՐԱՎՈՐՄԱՆ ՀԻՄՈՒՆՔՆԵՐ» դասընթաց

այլ ոլորտներից դեպի տեխնոլոգիական ոլորտ սկսնակների համար

edu2020.am

ԴԱՍ #13

ՀԱՅ-ՌՈՒՍԱԿԱՆ ՀԱՄԱԼՍԱՐԱՆ

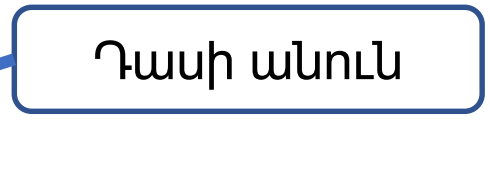
ՀԱՅԱՍՏԱՆԻ ՀԱՆՐԱՊԵՏՈՒԹՅԱՆ
ԲԱՐՁՐ ՏԵԽՆՈԼՈԳԻԱԿԱՆ
ԱՐԴՅՈՒՆԱԲԵՐՈՒԹՅԱՆ ՆԱԽԱՐԱՐՈՒԹՅՈՒՆ

Օբյեկտա-կողմնորոշված ծրագրավորում

- Պրոցեդուրային ծրագրավորման դեպքում խնդիրը լուծվում է ֆունկցիաների (կամ պրոցեդուրաների) միջոցով, որոնք կատարում են գործողություններ տվյալների հետ
- Օբյեկտա-կողմնորոշված ծրագրավորման դեպքում ստեղծվում են կառուցվածքներ, որոնք պարունակում են և՛ տվյալները, և ֆունկցիաները
- Օբյեկտա-կողմնորոշված ծրագրավորման առավելությունները՝
  - Հնարավորություն է տալիս մոդելավորել իրական օբյեկտներ և կատարել գործողություններ դրանց հետ
  - Տվյալները և դրանց մշակող ֆունկցիաները գտնվում են մի կառուցվածքի մեջ (class)
  - Կոդի կրկնություններից խուսափում

Դաս

- Դասը (class) նախատեսված է նոր տվյալների տիպ ստեղծելու համար, որը ներառում է տվյալներ և ֆունկցիաներ

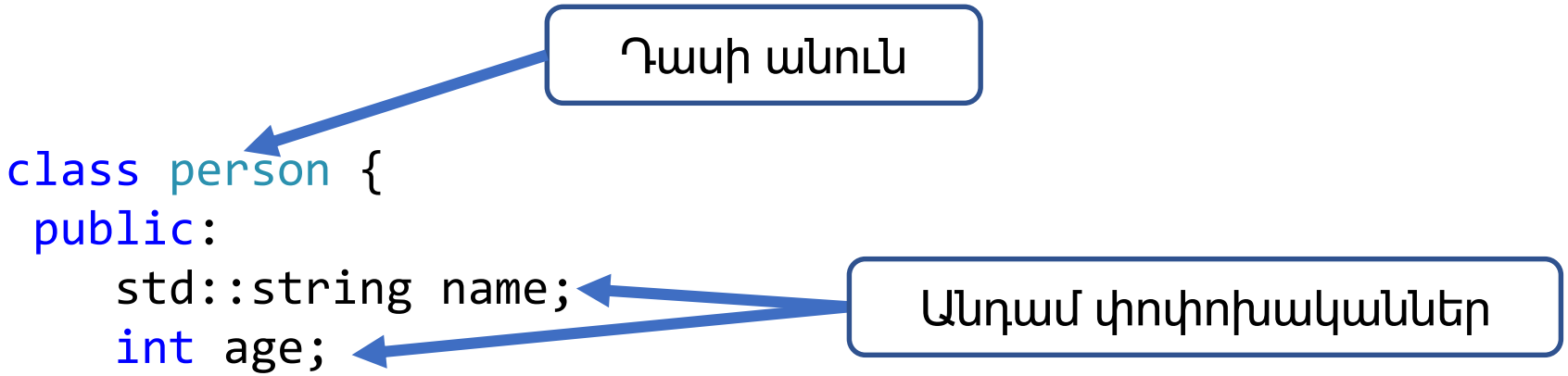



```
class person {
 public:
    std::string name;
    int age;

    void printInfo() {
      std::cout << name << " " << age << "\n";
    }
};
```

https://repl.it/@HaykAslanyan/classperson

Դաս

- Դասը (class) նախատեսված է նոր տվյալների տիպ ստեղծելու համար, որը ներառում է տվյալներ և ֆունկցիաներ

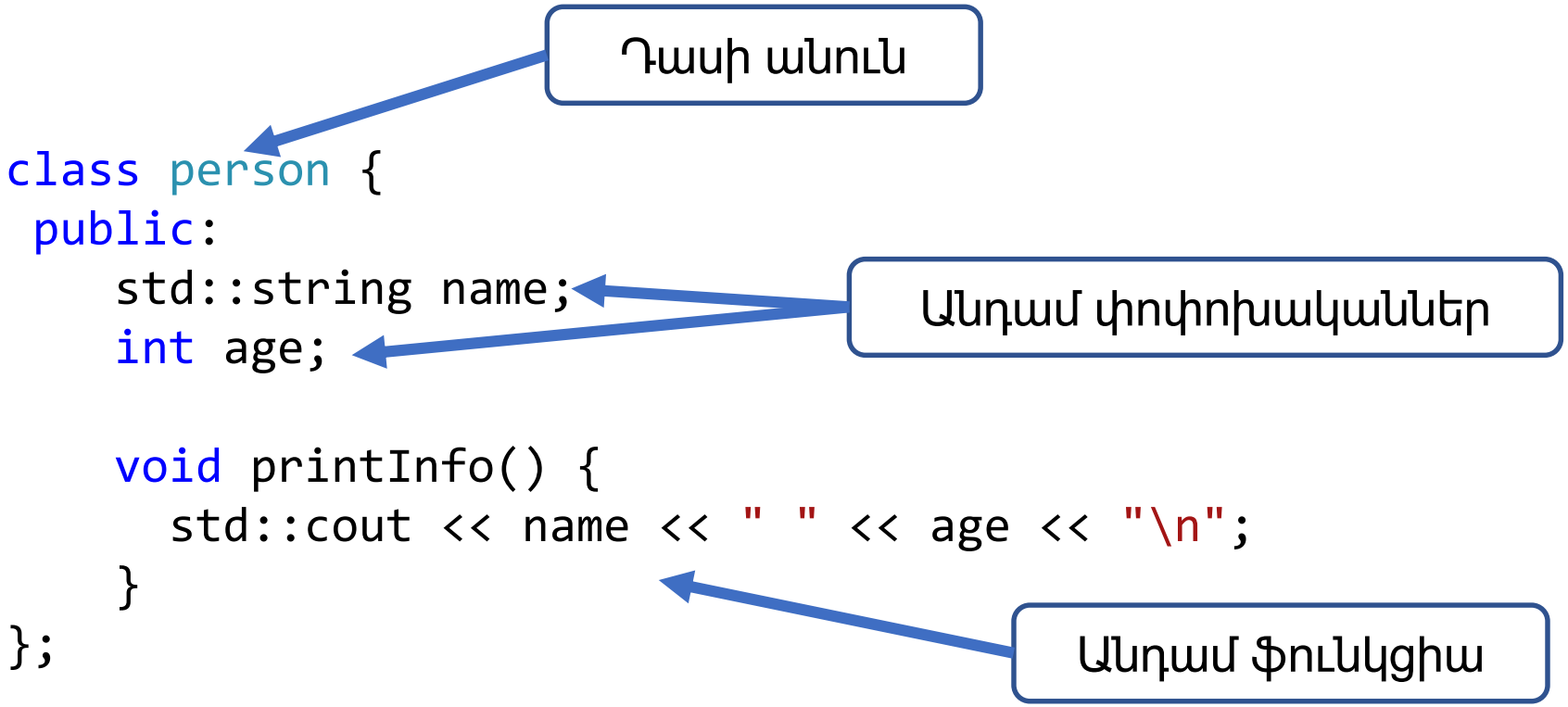



```
class person {
 public:
    std::string name;
    int age;

    void printInfo() {
      std::cout << name << " " << age << "\n";
    }
};
```

https://repl.it/@HaykAslanyan/classperson

Դաս

- Դասը (class) նախատեսված է նոր տվյալների տիպ ստեղծելու համար, որը ներառում է տվյալներ և ֆունկցիաներ

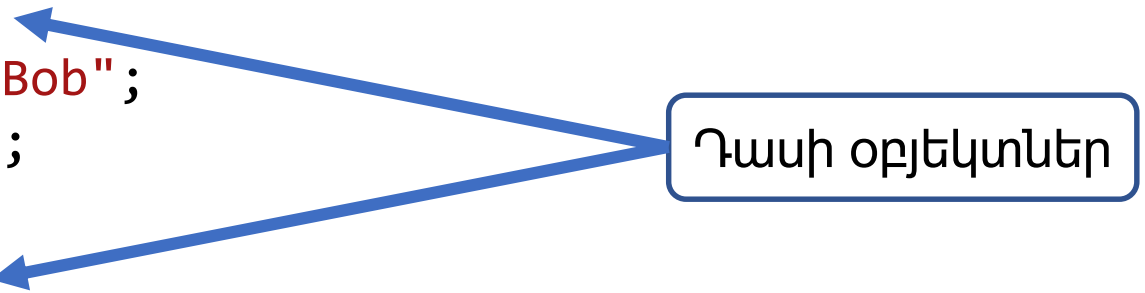


https://repl.it/@HaykAslanyan/classperson

Դաս

- Կարող են լինել տարբեր մարդիկ տարբեր անուններով և տարիքի, սակայն նրանք ունեն ընդհանուր հատկություններ
- Դասի օբյեկտը տվյալ դասի տիպ ունեցող փոփոխական է

```
int main() {
    person p1;
    p1.name = "Bob";
    p1.age = 25;

    person p2;
    p2.name = "Alex";
    p2.age = 50;

    p1.printInfo();
    p2.printInfo();
}
```



https://repl.it/@HaykAslanyan/classperson

Դաս

- Երբ դասը սահմանվում է, հիշողություն չի հատկացվում, այն հատկացվում է օբյեկտ ստեղծելիս

```
int main() {
    person p1;
    p1.name = "Bob";
    p1.age = 25;

    person p2;
    p2.name = "Alex";
    p2.age = 50;

    p1.printInfo();
    p2.printInfo();
}
```

Օբյեկտի տվյալին կարող ենք դիմել ‘.’-ի միջոցով

Ֆունկցիաները նույնպես կանչում ենք ‘.’-ի միջոցով

https://repl.it/@HaykAslanyan/classperson

Դաս

```
int main() {
    person p1;
    p1.name = "Bob";
    p1.age = 25;

    person p2;
    p2.name = "Alex";
    p2.age = 50;

    p1.printInfo();
    p2.printInfo();

    person* pointerOnPerson = &p1;
    pointerOnPerson->printInfo();
}
```

Կարող ենք ունենալ ցուցիչ դասի օբյեկտի վրա

Ցուցիչի միջոցով ֆունկցիաներին և տվյալներին կարող ենք դիմել ‘->’ միջոցով

Ինկապսուլյացիա

- Ինկապսուլյացիան (encapsulation) օբյեկտա-կողմնորոշված ծրագրավորման առաջին սկզբունքն է
- Այն սահմանվում է որպես տվյալներն ու դրանք մշակող ֆունկցիաները իրար հետ կապող սկզբունք
- Ինկապսուլյացիայի սկզբունքը բերում է տվյալները «ծածկելուն»

Ինկապսուլյացիա

Հասանելիության սպեցիֆիկատորներ

```
class person {
 private:
    std::string name;
    int age;

 public:
    person(std::string n, int a) : name(n), age(a) {
      std::cout << "person class constructor\n";
    }
    ~person() {
      std::cout << "person class destructor\n";
    }
    void printInfo() {
      std::cout << name << " " << age << "\n";
    }
};
```

```
int main() {
  person p1("Bob", 25);
  person p2("Alex", 50);

  p1.printInfo();
  p2.printInfo();

  person* pointerOnPerson = &p1;
  pointerOnPerson->printInfo();
}
```

https://repl.it/@HaykAslanyan/encapsulation

Ինկապսուլյացիա

```
class person {
 private:
    std::string name;
    int age;

 public:
    person(std::string n, int a) : name(n), age(a) {
      std::cout << "person class constructor\n";
    }
    ~person() {
      std::cout << "person class destructor\n";
    }
    void printInfo() {
      std::cout << name << "  " << age << "\n";
    }
};
```

Հասանելիության սպեցիֆիկատորներ

```
int main() {
  person p1("Bob", 25);
  person p2("Alex", 50);
  p1.name = "Jon"; // error

  p1.printInfo();
  p2.printInfo();

  person* pointerOnPerson = &p1;
  pointerOnPerson->printInfo();
}
```

Թարգմանության սխալ

https://repl.it/@HaykAslanyan/encapsulation

Հասանելիության սպեցիֆիկատորներ

- **public** անդամներին և ֆունկցիաներին հնարավոր է դիմել դասից դուրս
- **private** անդամներին և ֆունկցիաներին հնարավոր է դիմել միայն դասի մեջ [1]
- **protected** անդամներին և ֆունկցիաներին հնարավոր է դիմել միայն դասի և նրանից ժառանգած դասերի մեջ [1]

[1] Բացառություն են կազմում ընկեր ֆունկցիաները և դասերը

Կառուցիչ

- Դասի կառուցիչը (constructor) հատուկ ֆունկցիա է, որը կանչվում է այդ դասի օբյեկտ ստեղծելուց
- Կառուցիչն ունի նույն անունն ինչ դասը


```
class person {
 private:
    std::string name;
    int age;

 public:
    person(std::string n, int a) : name(n), age(a) {
      std::cout << "person class constructor\n";
    }
    ~person() {
      std::cout << "person class destructor\n";
    }
    void printInfo() {
      std::cout << name << " " << age << "\n";
    }
};
```

https://repl.it/@HaykAslanyan/encapsulation

Կառուցիչ

- Դասի կառուցիչը (constructor) հատուկ ֆունկցիա է, որը կանչվում է այդ դասի օբյեկտ ստեղծելուց
- Կառուցիչն ունի նույն անունն ինչ դասը

```
class person {
 priva
    st
    int age;

 public:
    person(std::string n, int a) : name(n), age(a) {
      std::cout << "person class constructor\n";
    }
    ~person() {
      std::cout << "person class destructor\n";
    }
    void printInfo() {
      std::cout << name << " " << age << "\n";
    }
};
```

Կառուցիչի կանչ

```
int main() {
  person p1("Bob", 25);
  person p2("Alex", 50);

  p1.printInfo();
  p2.printInfo();

  person* pointerOnPerson = &p1;
  pointerOnPerson->printInfo();
}
```

https://repl.it/@HaykAslanyan/encapsulation

Կառուցիչ

- Դասի կառուցիչը (constructor) հատուկ ֆունկցիա է, որը կանչվում է այդ դասի օբյեկտ ստեղծելուց
- Կառուցիչն ունի նույն անունն ինչ դասը

```
int main() {
  person *P = new person("Bob", 25);

  delete P;
}
```

```
class person {
 priva
    st
    int age;

 public:
    person(std::string n, int a) : name(n), age(a) {
      std::cout << "person class constructor\n";
    }
    ~person() {
      std::cout << "person class destructor\n";
    }
    void printInfo() {
      std::cout << name << " " << age << "\n";
    }
};
```

Կառուցիչի կանչ

https://repl.it/@HaykAslanyan/encapsulation

Լռությամբ կառուցիչ

- Լռությամբ կառուցիչն այն կառուցիչն է, որը կարող է կանչվել առանց արգումենտների
- Եթե դասի համար ոչ մի կառուցիչ սահմանված չէ, կոմպիլյատորը կստեղծի լռությամբ կառուցիչ
  - Բացառություն են կազմում այն դասերը, որոնք որպես անդամ ունեն հղում կամ կոնստանտ

Լռությամբ կառուցիչ

```
#include <iostream>

class Date {
 public:
  int year;
  int month;
  int day;
  Date() : year(2020), month(1), day(1) {}
};

int main() {
  Date date1;
  Date date2 = Date();
  std::cout << date1.year << " " << date1.month << " " << date1.day << "\n";
  std::cout << date2.year << " " << date2.month << " " << date2.day << "\n";
}
```

Լռությամբ կառուցիչի կանչ

https://repl.it/@HaykAslanyan/Defaultconstructor1

Լռությամբ կառուցիչ

```cpp
#include <iostream>

class Date {
 public:
  int year;
  int month;
  int day;
  // Date() : year(2020), month(1), day(1) {}
};

int main() {
  Date date1;
  Date date2 = Date();
  std::cout << date1.year << "\n"; // Bad code
  std::cout << date2.year << "\n"; // OK
  date1.month = 1;
  std::cout << date1.month; // OK
}
```

Կոմպիլյատորը կստեղծի լռությամբ կառուցիչ

year, month, day անդամների արժեքները կլինեն անորոշ

year, month, day անդամների արժեքները կլինեն 0

https://repl.it/@HaykAslanyan/defaultconstructor2

Փլուզիչ

- Դասի փլուզիչը (destructor) հատուկ ֆունկցիա է, որը կանչվում է այդ դասի օբյեկտը ջնջվելուց


```
class person {
 private:
    std::string name;
    int age;

 public:
    person(std::string n, int a) : name(n), age(a) {
      std::cout << "person class constructor\n";
    }
    ~person() {
      std::cout << "person class destructor\n";
    }
    void printInfo() {
      std::cout << name << " " << age << "\n";
    }
};
```

https://repl.it/@HaykAslanyan/encapsulation

Փլուզիչ

- Դասի փլուզիչը (destructor) հատուկ ֆունկցիա է, որը կանչվում է այդ դասի օբյեկտը ջնջվելուց

```
class person {
 private:
    std::string name;
    int age;

 public:
    person(std::string n, int a) : name(n), age(a) {
      std::cout << "person class constructor\n";
    }
    ~person() {
      std::cout << "person class destructor\n";
    }
    void printInfo() {
      std::cout << name << " " << age << "\n";
    }
};
```

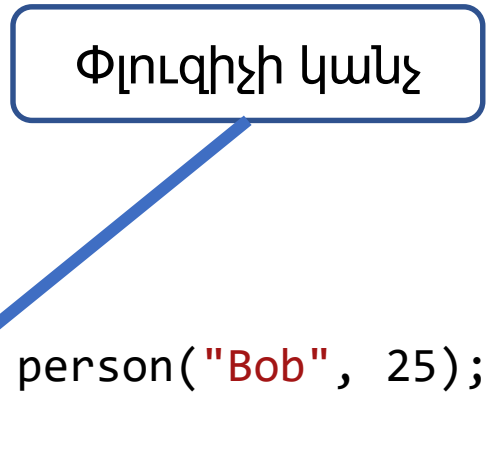
```
int main() {
  person p1("Bob", 25);
  person p2("Alex", 50);

  p1.printInfo();
  p2.printInfo();

  person* pointerOnPerson = &p1;
  pointerOnPerson->printInfo();
}
```

Փլուզիչի կանչ

https://repl.it/@HaykAslanyan/encapsulation

Փլուզիչ

- Դասի փլուզիչը (destructor) հատուկ ֆունկցիա է, որը կանչվում է այդ դասի օբյեկտը ջնջվելուց




```
int main() {
  person *P = new person("Bob", 25);

  delete P;
}
```

```
class person {
 private:
    std::string name;
    int age;

 public:
    person(std::string n, int a) : name(n), age(a) {
      std::cout << "person class constructor\n";
    }
    ~person() {
      std::cout << "person class destructor\n";
    }
    void printInfo() {
      std::cout << name << " " << age << "\n";
    }
};
```

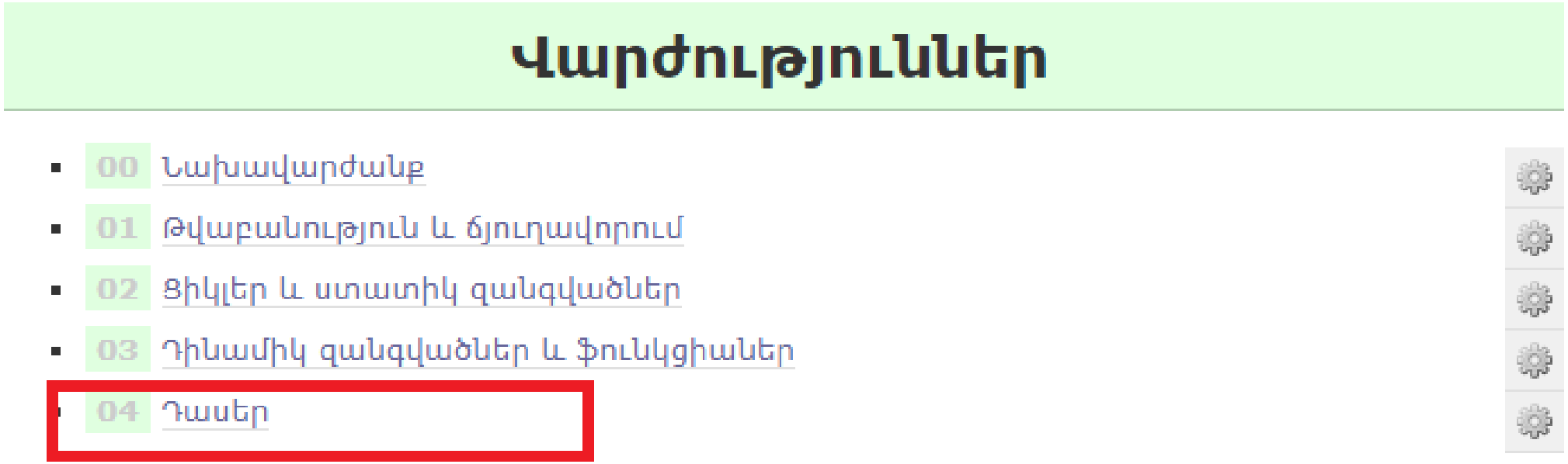
https://repl.it/@HaykAslanyan/encapsulation

Դաս

- Գրել point դասը դեկարտյան կոորդինատային համակարգում կետը մոդելավորելու համար
- Գրել rectangle դասը ուղղանկյուն մոդելավորելու համար
- Գրել car դասը ավտոմեքենաներ մոդելավորելու համար

Տնային աշխատանք

Տնային աշխատանք 1-8



Շնորհակալություն. Հարցե՞ր