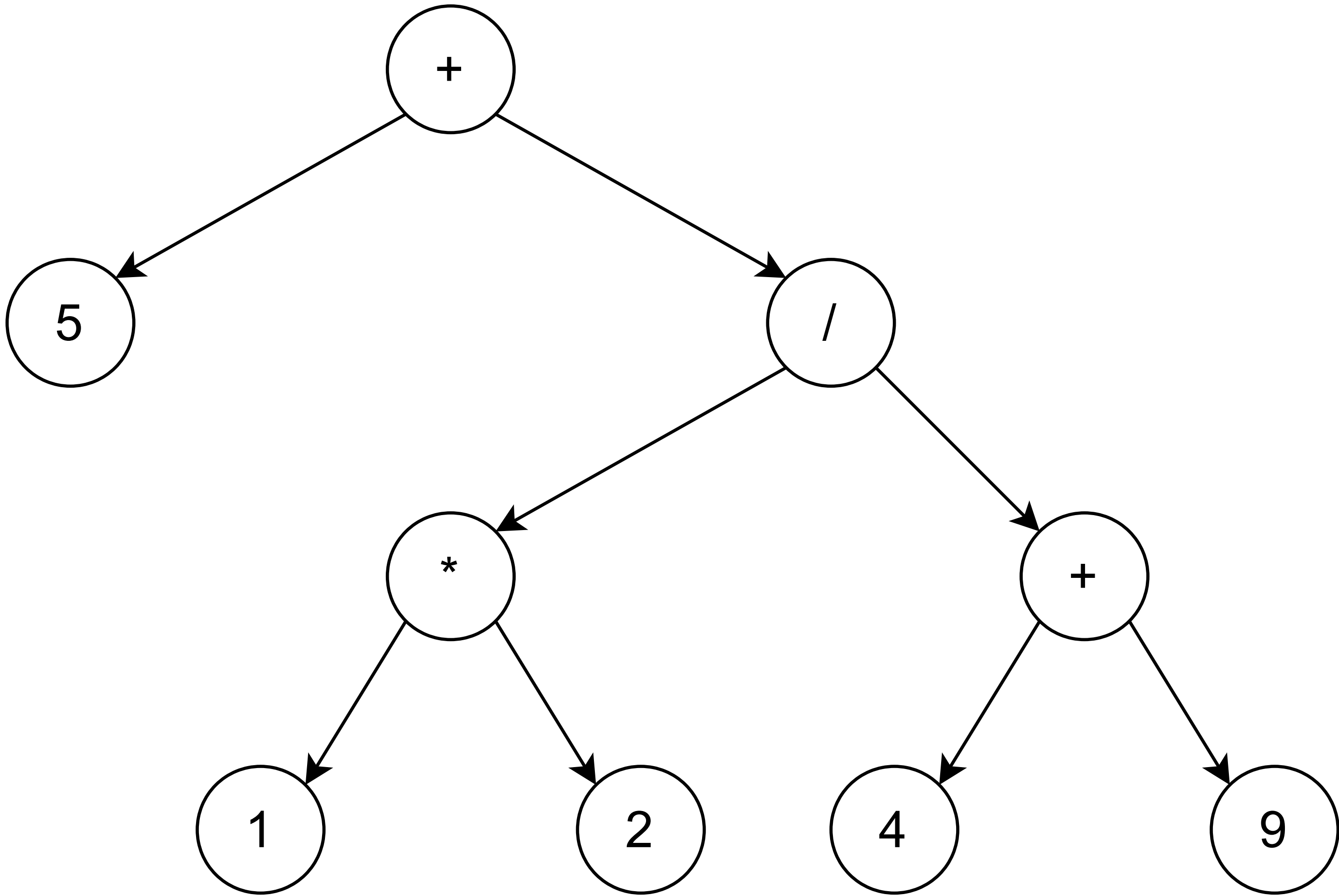
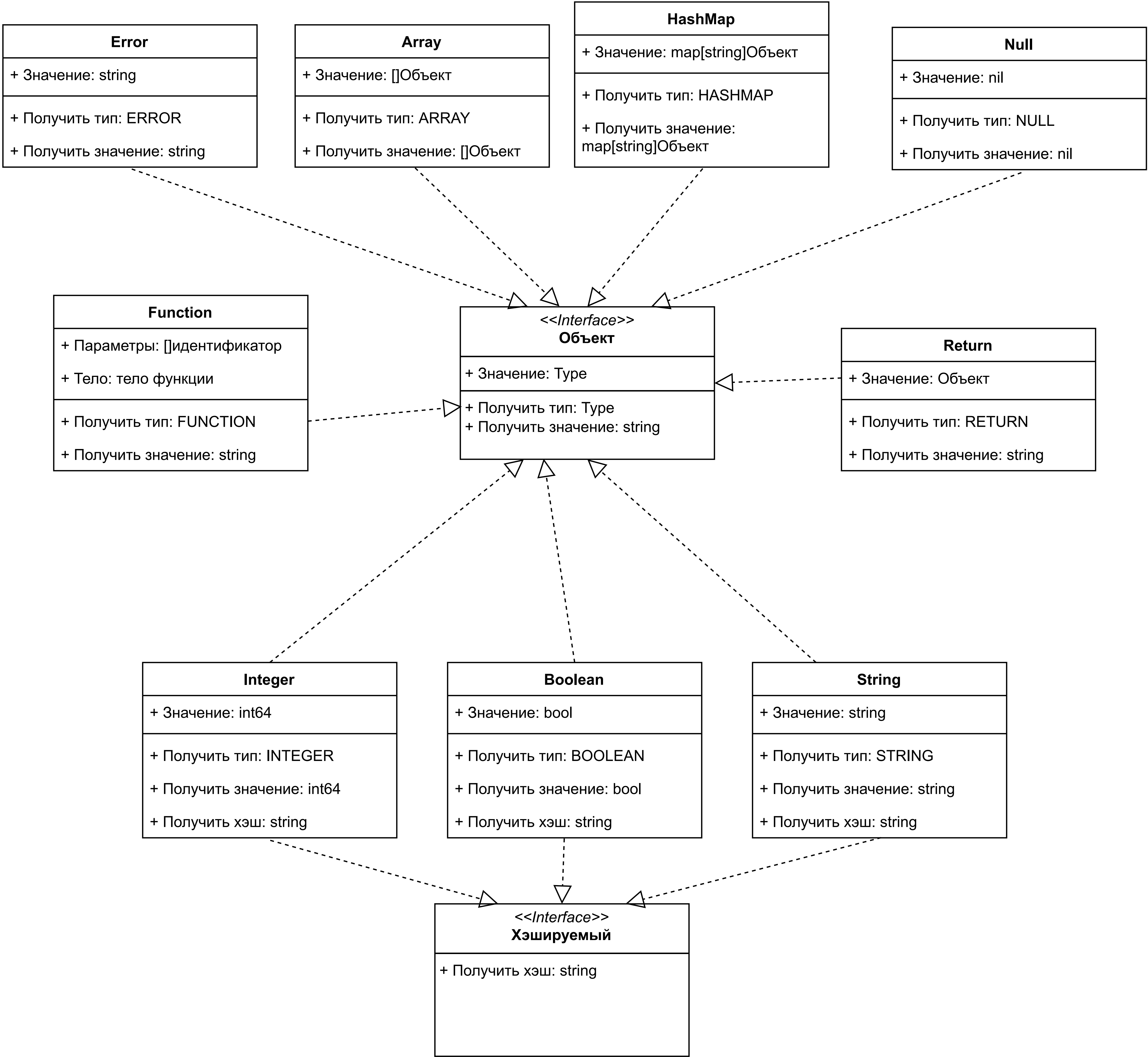


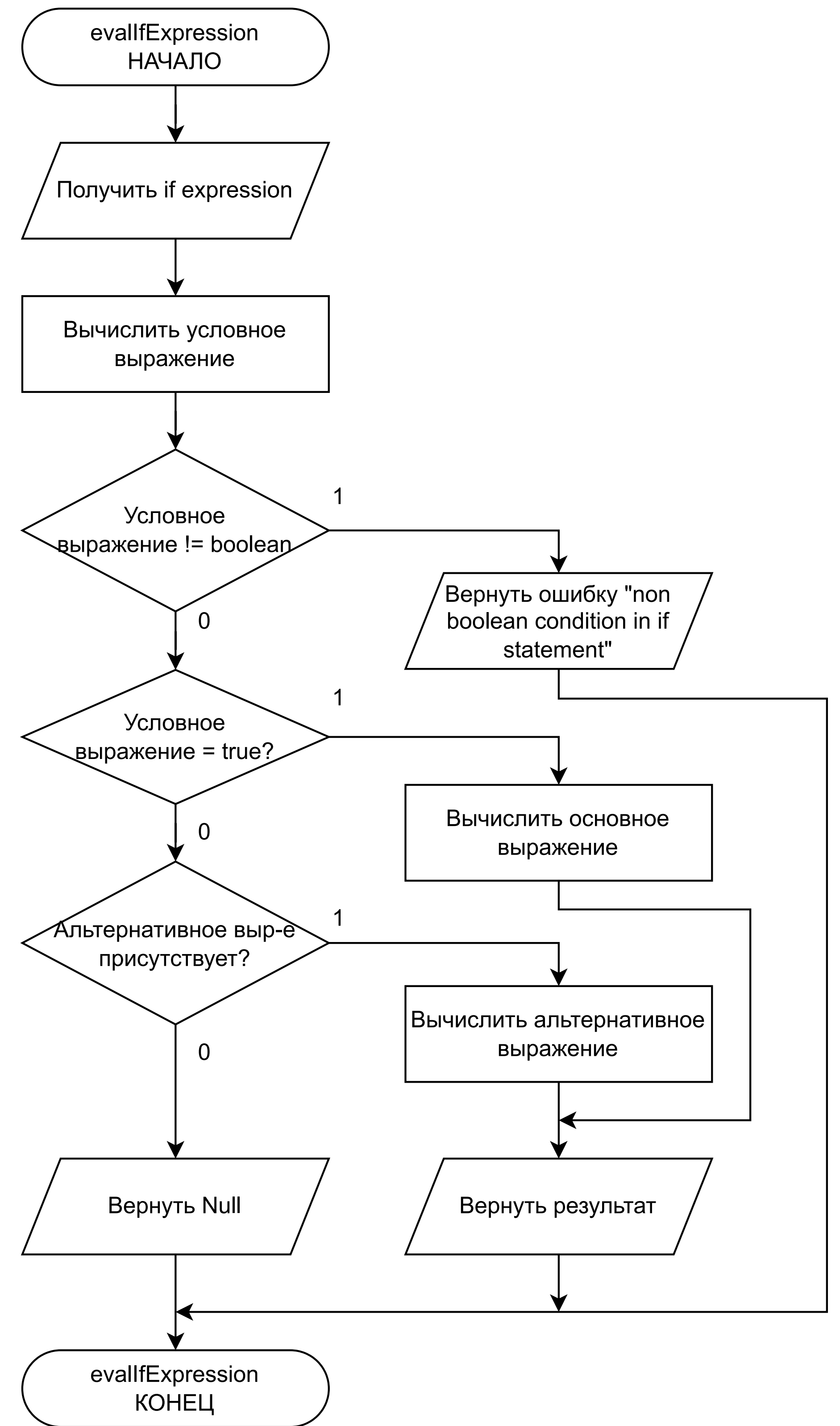
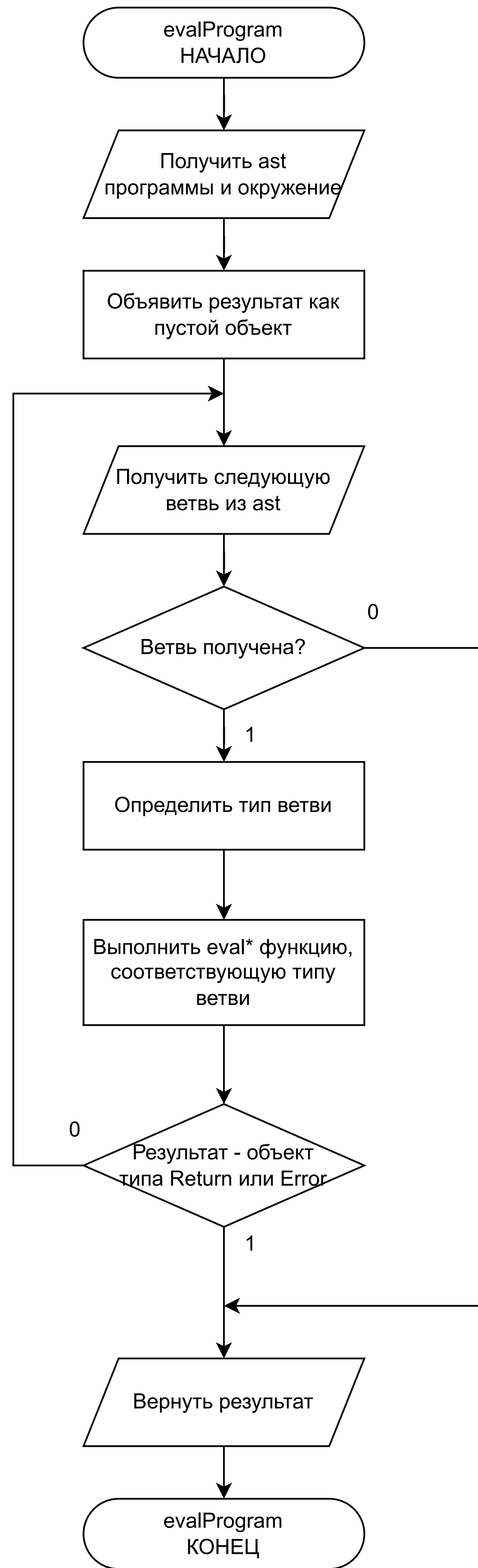
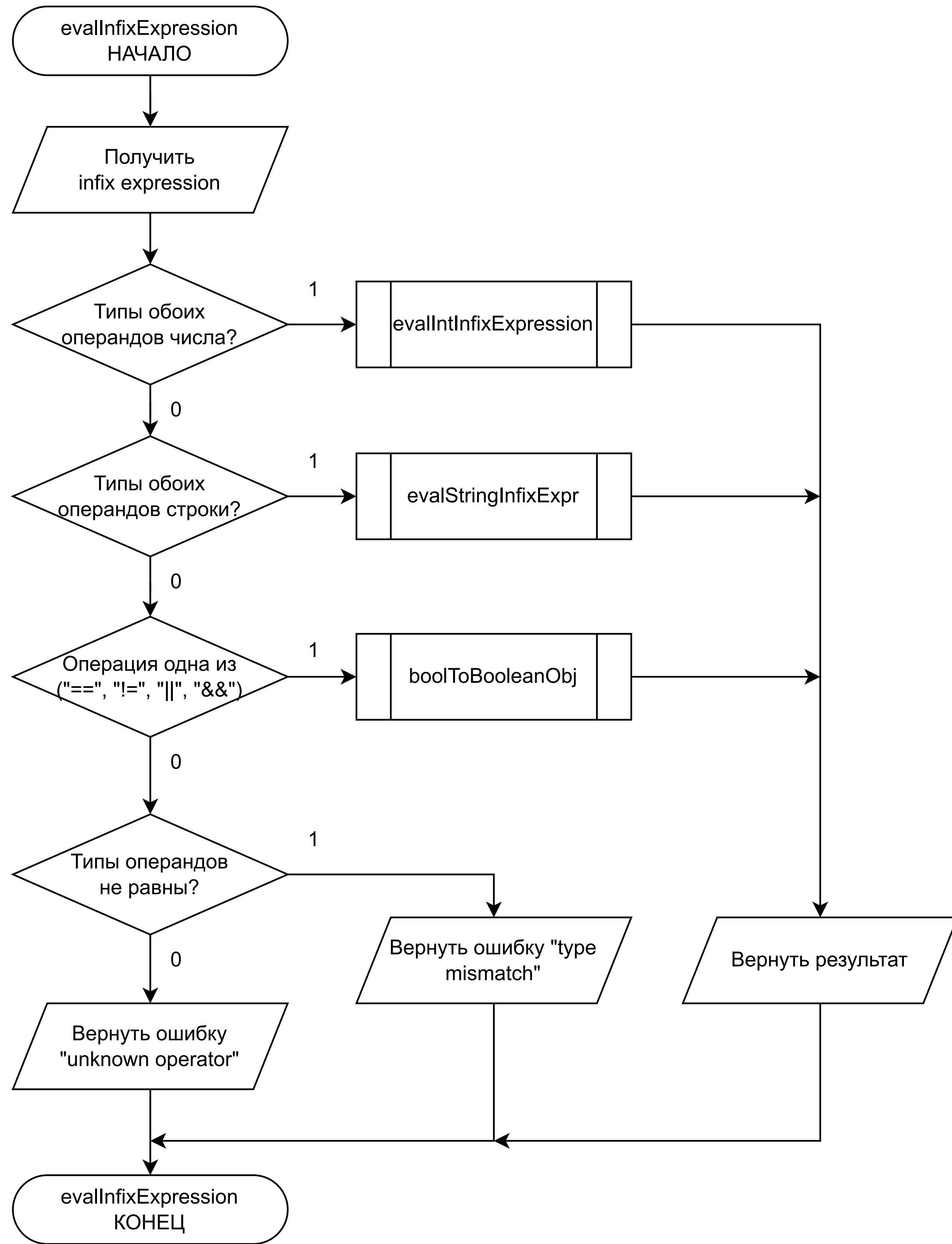
```
(*ast.ExpressionStatement)({
  Token: (token.Token) {
    |   Type: (string) (len=3) "INT",
    |   Literal: (string) (len=1) "5"
    | },
  Expression: (*ast.InfixExpression)({
    Token: (token.Token) {
      |   Type: (string) (len=1) "+",
      |   Literal: (string) (len=1) "+"
      | },
    Left: (*ast.IntegerLiteral)({
      Token: (token.Token) {
        |   Type: (string) (len=3) "INT",
        |   Literal: (string) (len=1) "5"
        | },
      Value: (int64) 5
    }),
    Operator: (string) (len=1) "+",
    Right: (*ast.InfixExpression)({
      Token: (token.Token) {
        |   Type: (string) (len=1) "/",
        |   Literal: (string) (len=1) "/"
        | },
      Left: (*ast.InfixExpression)({
        Token: (token.Token) {
          |   Type: (string) (len=1) "**",
          |   Literal: (string) (len=1) "**"
          | },
        Left: (*ast.IntegerLiteral)({
          Token: (token.Token) {
            |   Type: (string) (len=3) "INT",
            |   Literal: (string) (len=1) "1"
            | },
            Value: (int64) 1
          }),
          Operator: (string) (len=1) "**",
          Right: (*ast.IntegerLiteral)({
            Token: (token.Token) {
              |   Type: (string) (len=3) "INT",
              |   Literal: (string) (len=1) "2"
              | },
              Value: (int64) 2
            })
          }),
          Operator: (string) (len=1) "/",
          Right: (*ast.InfixExpression)({
            Token: (token.Token) {
              |   Type: (string) (len=1) "+",
              |   Literal: (string) (len=1) "+"
              | },
            Left: (*ast.IntegerLiteral)({
              Token: (token.Token) {
                |   Type: (string) (len=3) "INT",
                |   Literal: (string) (len=1) "4"
                | },
                Value: (int64) 4
              }),
              Operator: (string) (len=1) "+",
              Right: (*ast.IntegerLiteral)({
                Token: (token.Token) {
                  |   Type: (string) (len=3) "INT",
                  |   Literal: (string) (len=1) "9"
                  | },
                  Value: (int64) 9
                })
              })
            })
          })
        })
      })
    })
  })
})
```

app/app.go:45 (5 + ((1 * 2) / (4 + 9)))

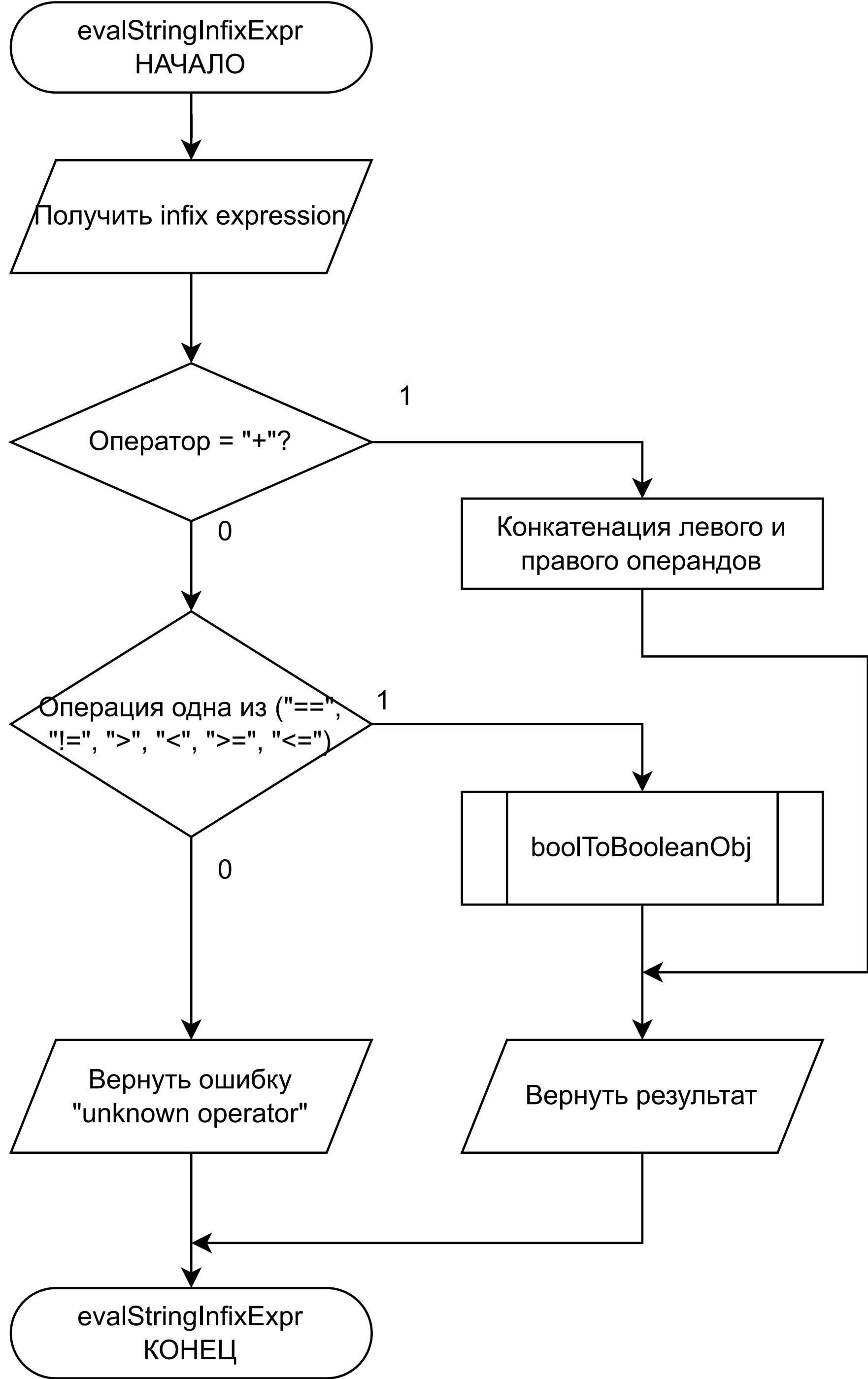
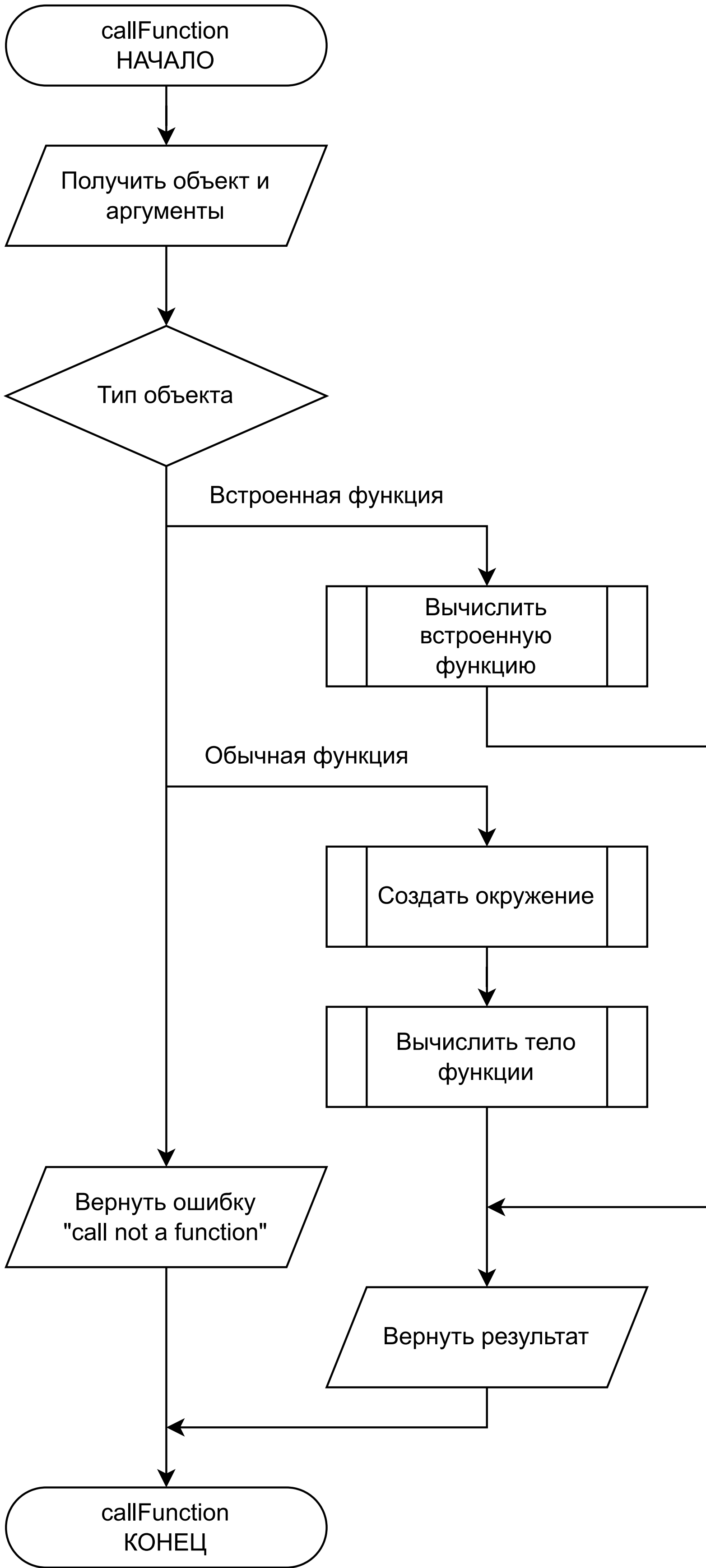
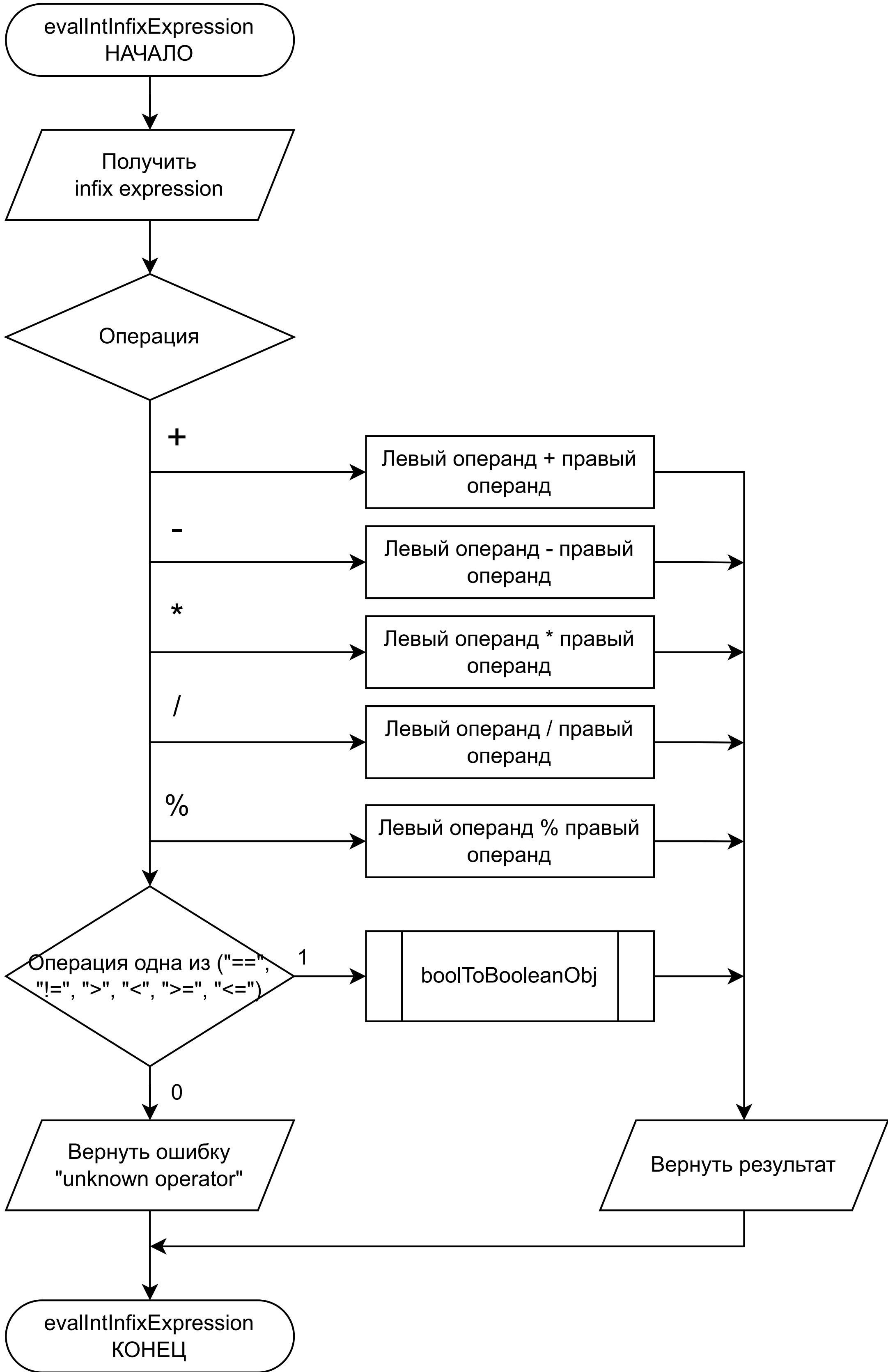


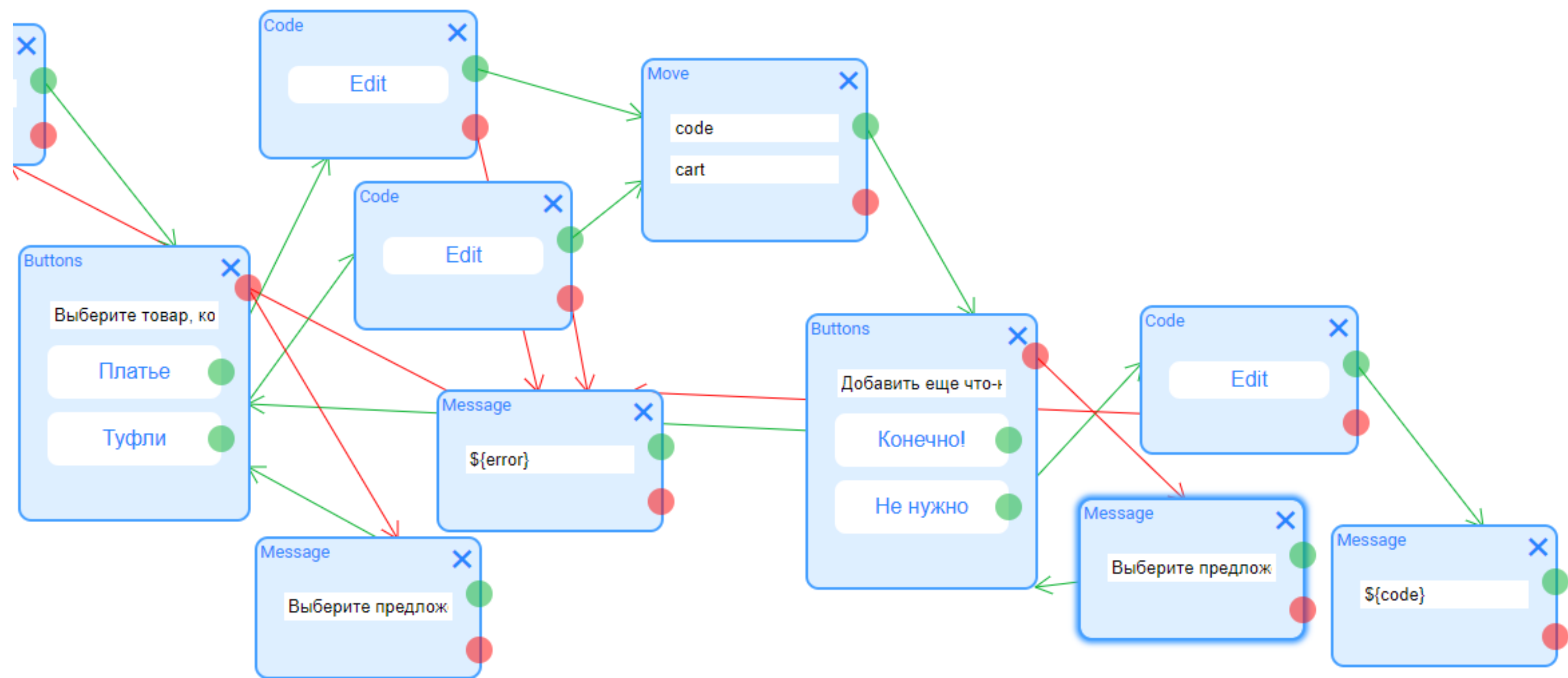
						ТПЖА 09.03.01.331-04 ДПЛ					
						Абстрактное синтаксическое дерево			Лит.	Масса	Масштаб
Изм.	Лист	№ докум.	Подп.	Дата							
Разраб.		Крючков									
Пров.		Долженкова									
Т. контр.									Лист 1		Листов 1
Н. контр.		Скворцов							Кафедра ЭВМ Группа ИВТ-41		
Утв.		Долженкова									





					ТПЖА 09.03.01.331-06 ДПЛ					
					Схемы алгоритмов семантического анализатора		Лит.		Масса	Масштаб
Изм.	Лист	№ докум.	Подп.	Дата			Лист 1		Листов 1	
Разраб.		Крючков								
Прое.		Долженкова								
Т. контр.										
Н. контр.		Скворцов								
Уте.		Долженкова								





```
pprice = cart["Платье"] * price["Платье"];
tprice = cart["Туфли"] * price["Туфли"];

t = ""
if(cart["Платье"] > 0) {
    t = t + "- Платье: " + intToString(cart["Платье"]) + " ед.: " + intToString(pprice) + " р. \n"
}

if(cart["Туфли"] > 0) {
    t = t + "- Туфли: " + intToString(cart["Туфли"]) + " ед.: " + intToString(tprice) + " р. \n"
}

if(cart["Туфли"] == 0 && cart["Платье"] == 0) {
    t = "Корзина пуста\n"
}

s = "Ваша корзина:\n " + t + "\n Общая стоимость: " + intToString(pprice + tprice) + " р."

s
```

Save Cancel

/start 0:01 ✓✓

Здравствуйте 0:01

Выберите товар, который хотите добавить в корзину 0:01

Платье 0:48 ✓✓

Добавить еще что-нибудь? 0:48

Конечно! 0:48 ✓✓

Выберите товар, который хотите добавить в корзину 0:48

Платье 0:48 ✓✓

Добавить еще что-нибудь? 0:48

Конечно! 0:48 ✓✓

Выберите товар, который хотите добавить в корзину 0:48

Туфли 0:48 ✓✓

Добавить еще что-нибудь? 0:48

Не нужно 0:48 ✓✓

Ваша корзина:
- Платье: 2 ед.: 4500 р.
- Туфли: 1 ед.: 1200 р.

Общая стоимость: 5700 р. 0:48