Ответы на вопросы экзамена по курсу «Языки программирования» 19.01.2015

В ответах курсивом выделены необязательные пояснения, которые можно опустить (особенно на экзамене)

Вариант 2

Задача 2-1

В каких языках из перечисленных ниже можно реагировать на ошибки переполнения, связанные с целочисленной арифметикой? Для каждого языка приведите пример соответствующих конструкций.

Си++, Модула-2, Java, Delphi, C#, Оберон, Python, JavaScript

Ответ

Речь идет об ошибках, связанных с выполнением арифметических операций, например, при сложении. Присваивание значений других типов (выходящих за пределы диапазона значений) не имеет никакого отношения собственно к арифметике.
Также речь не идет об особенностях конкретных реализаций (например, опциях конкретных компиляторов). Речь идет о стандартных языковых конструкциях, которые позволяют включать реакцию на ошибки арифметического переполнения.
Из перечисленных выше языков такой особенностью обладает язык C#. Он дает возможность проконтролировать корректность результата целочисленных операций. Для этого служит конструкция `checked`:

```
checked  {
  for (uint i = 256; i >= 0; i--) cnt++;
}
```

В блоке, следующем за ключевым словом `checked`, все целочисленные операции проверяются на корректность, поэтому при возникновении переполнения (при попытке вычесть 1 из нулевого беззнакового значения) генерируется ошибка. Разумеется, время выполнения операций растет, поэтому по умолчанию операции не контролируются.

Среагировать на ошибку можно с помощью механизма обработки исключений.
В других языках либо отсутствуют стандартные средства контроля, лиюо вычисления при выходе за границу диапазона переходят к расширенным типам данных и т.п.

Задача 1-2

Напишите спецификацию шаблона `template <int N> struct ArraySum {....};` на языке C++ так, чтобы для любого неотрицательного целого константного значения `n` значение `ArraySum<n>::result` было равно сумме первых n элементов массива Arr, описанного следующим образом:

```
static constexpr int Arr[] = {1,5,-3,.../* и так далее */...};
```

Ответ

Поскольку речь идет о шаблоне, то подразумевается, что вычисления производятся статически (во время трансляции). Текст программы(в ответе достаточно было описания шаблонов, ошибки задания n не проверяются):

```
#include <iostream>

static constexpr int a[] = {1,2,3,4};

template <int n> struct ArraySum
{
    enum {r = ArraySum<n-1>::r + a[n-1]};
};

template <> struct ArraySum<1>
{
    enum {r = a[0]};
};

int main()
{
    std::cout << ArraySum<10>::r << "\n";
    return 0;
}
```

Задача 1-3

Можно ли в языке C++ определить тело абстрактной функции? Если можно, то приведите пример контекста, в котором вызывается это тело.

Ответ

Несмотря на то, что обычно абстрактные функции в языках программирования не имеют тела (*не путайте тело функции с функцией-заместителем — это разные понятия*), абстрактные функции в языке C++ *(называемые там чистыми виртуальными функциями),* тем не менее, могут иметь тело.
Возникает вопрос, а какой смысл в теле, если функция должна быть замещена в наследниках, объекты абстрактного класса создать невозможно, а при вызове функции из наследника будет работать механизм замещения. Так что, на первый взгляд, тело абстрактной функции вызвать нельзя.
Существуют контексты, когда тело абстрактной функции может вызвано (отсюда и возможность его определения).
Например, вызов виртуальной функции (прямо или косвенно) из конструктора абстрактного класса:

```
class Test
{
public:
	Test()
	{
		f();
	}
	virtual void f() = 0;
};
// вот и тело абстрактной функции
void Test::f()
{
	std::cout << "Hello, abstract!\n";
}

class Conc : public Test
{
```

```
public:
	void f()
	{
	}
};

int main()
{
	Conc t;

	return 0;
}
```

При создании объекта класса `Conc` сработает конструктор класса `Test`, и оттуда будет вызвана абстрактная функция `Test::f`
Кроме того, можно вызвать абстрактную функцию напрямую:
`Conc t;  t.Test::f();`
При отсутствии тела (нормальный случай), вызывается функция из библиотеки, которая аварийно завершает выполнение программы (отсюда и диагностика во время выполнения). Если программист вдруг захотел отменить это поведение (зачем — не будем обсуждать), то он и использует этот механизм.

Задача 1-4

Напишите определение функции-генератора с именем `NN` на языке Python так, чтобы следующий фрагмент программы выдавал первые 100 квадратов целых чисел (1,4,9,16,.....):

```
for x in NN(100):
  print(x)
```

Ответ

```
def NN(n):
     for in range(1,n+1):
          yield i*i
```

Задача 1-5

Дайте общее определение операции преобразования типа в языке программирования. Напишите пример пользовательского переопределения операции преобразования типа в языке C++ (определение операции и ее вызов). Есть ли аналогичные средства в языке C#? Если есть, то приведите пример на этом языке, аналогичный приведенному для C++.

Ответ

Операция преобразования типа – это функция:
F: T1 => T2. Она создает по объекту типа T1 значение типа T2.
Базисная (то есть встроенная в язык) операция преобразования применима к некоторым базисным типам T1,T2. Для C/C++ T1 и T2 могут быть, например, любыми арифметическими типами. Операции преобразования могут быть явными (явно указываются — как правило, это можно для любой допустимой операции преобразования) и неявными (подставляются компилятором — когда и как — зависит от языка, например, арифметические преобразования в C/C++ могут быть неявными).
В некоторых языках семантика операции преобразования может быть расширена на пользовательские типы данных.

В C++ допустимо переопределять операции преобразования, если:

1). T1 – любой тип данных, а T2 – класс. В этом случае преобразование задается конструктором преобразования класса T2:

```
class T2 {
public:
 T2(int x); // T1 - int
 .....
};
Если перед конструктором стоит explicit, то этот
конструктор задает только явную операцию преобразования, в
противном случае преобразование -неявное.
```

2). T1 – класс, а T2 — любой тип данных. В этом случае преобразование задается функцией преобразования класса T2:

```
class T1 {
public:
     operator int();// T2 - int
     ...
};
В этом случае операция преобразования — неявная.
```

В языке C# операции преобразования могут определяться только для пользовательских классов (но не встроенных типов и классов), при этом они задаются как статические функции члены либо класса T1, либо класса T2:

```
class T1{
     ....
}
class T2{
     public static operator T2(T1 t){ .... }
}
```

Перед операцией преобразования может стоять ключевое слово `explicit` или `implicit` (явная или неявная операция). По умолчанию операция явная.

Поэтому для примера выше можно писать:

```
T1 t = new t();
T2 tt = (T2)t;
```

Но нельзя:

```
T1 t = new t();
T2 tt = t; // ошибка: есть только явная операция
```

Задача 1-6

Дайте определение и пример использования анонимной (лямбда-функции) в языке C#. В каких языках из перечисленных ниже есть аналогичное понятие? Для каждого такого языка приведите пример.

JavaScript, Ада, Си++, Оберон, Оберон-2

Ответ

Лямбда-функция — это безымянная функция, тело которой — выражение, которое возвращается как результат функции:

```
arguments => expression
```

Выражение может включать в себя не только аргументы, но и доступные переменные из окружающих областей видимости, которые «захватываются» лямбда-функцией и составляют так называемое «замыкание».

Пример:

```
var f = x=>x*x;
int x = f(5); // = 25
double d = f(1.717); // что-то около 3.0
```

Лямбда-функции есть также (из перечисленных в условии языков) в JavaScript,Си++.

JavaScript:
```
f = function(x) { return x*x; }
```

Си++:
```
// определим лямбда-функцию и немедленно применим ее к 5
auto twentyfive = [](int x) {return x*x; }(5);
```

Задача 1-7

Дайте определение понятия «объект, подобный массиву» в языке JavaScript. Перечислите его отличия от реального массива JavaScript.

Ответ

В JavaScript можно добавить к любому объекту свойство с именем `name`:
```
obj.name = "foo"
obj.name1 = 1
```
и т.д.
Это же можно сделать и так:
```
obj["name"] = "foo"
obj["name1"] = 1
```
Во втором случае имена полей могут вычисляться.
Поэтому если в качестве имен полей в операции индексирования задать целые значения (или строки, которые представляют собой целые значения), а также определить все поля с именами-индексами от 0 до какого-то N-1, а свойству `length` будет присвоено значение N, то по поведению этот объект будет практически не отличим от обычного массива.
Пример:
```
var obj = new Object()
for (i = 1; i < 100; i++)
     obj[i] = i*i
```
`obj` – не массив, но ведет себя как массив.
Основное его отличие от настоящего объекта `Array` – то, что у реального массива есть встроенное свойство `length`, изменение которого приводит к реальному изменению количества элементов:
```
var arr = {1,2,3,4,5} // 5 элементов
arr.length = 10 // добавилось в конец 5 неопределенных элементов
arr.length = 0 // все пропало!
```

Для `obj` из примера выше это, конечно, не работает.

Примером объекта, который ведет себя как массив, является значение свойства `arguments` в теле функции.

Задача 1-8

При ответе на эту задачу следует иметь ввиду то, что в Java приватные функции не замещаются в производных классах, так как они невидимы в них (см. также ответ на вопрос №3 из первого варианта). Поэтому приватная функция pf не будет замещаться в производных классах (неважно, замещается она публичной или приватной функцией — все равно не видна!).

Задача №8 имела 2 варианта. Первый:
Что будет напечатано в результате работы следующей программы на Java?

```
public class PLExamTest2014 {
  static void P(A a,B b) { a.g(); b.g(); }
  public static void main(String[] args) {
    P(new A(), new B());
   System.out.println("------------------------------");
    P(new B(), new C());
  }
}
class A {
  private void pf() {System.out.println("A.pf"); }
  public void f() { System.out.println("A.f");}
  public void g() { f(); pf(); }
}
class B extends A {
  public void pf() { System.out.println("B.pf");}
  public void f() { System.out.println("B.f");}
}
class C extends B {
  public void pf() { System.out.println("C.pf");}
  public void f() { System.out.println("C.f");}
}
```

Ответ

A.f
A.pf
B.f
A.pf

B.f
A.pf
C.f
A.pf

Второй вариант задачи №8:
Что будет напечатано в результате работы следующей программы на Java?

```
public class PLExamTest2014 {
  static void P(A a,B b) { a.g(); b.g(); }
  public static void main(String[] args) {
```

```
    P(new A(), new B());
    System.out.println("------------------------------");
    P(new B(), new C());
  }
}
class A {
  private void pf() {System.out.println("A.pf"); }
  public void f() { System.out.println("A.f");}
  public void g() { f(); pf(); }
}
class B extends A {
  private void pf() { System.out.println("B.pf");}
  public void f() { System.out.println("B.f");}
  public void g() { f(); pf(); }
}
class C extends B{
  public void pf() { System.out.println("C.pf");}
  public void f() { System.out.println("C.f");}
}
```

Ответ

A.f
A.pf
B.f
B.pf

B.f
B.pf
C.f
B.pf