Индусский код — Lurkmore

> **К вашему сведению!**
>
> В этой статье мы описываем само явление интусского кота, а не составляем списки всех возможных примеров. Ваше мнение о кривых реализациях и копипасту очередного ZOMG TEH HINDU CODE!!111! читать здесь никому не интересно, поэтому все правки с упоминанием «Пример № 100500» будут откачены, а их авторы — расстреляны на месте из реактивного говномета, for great justice!

«/** * For the brave souls who get this far: You are the chosen ones, * the valiant knights of programming who toil away, without rest, * fixing our most awful code. To you, true saviors, kings of men, * I say this: never gonna give you up, never gonna let you down, * never gonna run around and desert you. Never gonna make you cry, * never gonna say goodbye. Never gonna tell a lie and hurt you. **/ »

— *http://stackoverflow.com/questions/184618/*

«Хороший индеец — мёртвый индеец! »

— Слышано от senior software developer в международной компании после нескольких суток pefuckторинга

Индусский код (в среде программистов 80х также известен как Glitch) — в самом общем случае, это криво написанный, но каким-то удивительным образом работающий код. Индусский код написан наиболее неочевидным и неестественным из всех возможных способов. Именно этим он и отличается от быдлокода, который хотя бы капельку очевиден и сделан, хоть и по детсадовским, но по правилам.

По другой трактовке, **индусский код** — жаргонное нарицательное название для программного кода крайне низкого качества, использующего простые, но порочные принципы «copy-paste», «главное — завалить, а там запинаем», «шапкозакидательство». Также известен как «write only», или (в пику принципу «написано однажды — читается везде») — «написанное однажды — лучше не читать».

Индусский программист. «Пишу на C++ за еду!»

Почему именно индусский

```
xxx: странные в Индии программисты

xxx: глубокий индийский смысл кода intQuanty = intQuanty + 0;

yyy: буддистское созерцание присваивания
```

392993

В Индии с некоторых времен существует практика оценки производительности труда программиста на основе количества написанного кода. Чем больше кода, тем больше программист работает, и, следовательно, выше его оклад. Шустрые индусы быстро сообразили, как обманывать неквалифицированных заказчиков.

В результате чего можно встретить подобные шедевры (и это только цветочки!):

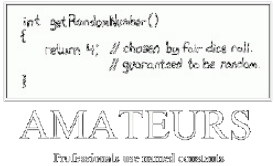
```
if (true) {
    // какой-то код
} else {
    // а вот тут чистый profit
}
```

Менее очевидный вариант:

```
if(var)
{
...
}
else if(!var)
{
...
}
else
{
// чистые деньги
}
```

Генеральный директор Microsoft с 2014 года — индус. Это должно было случиться

Аутентичный индусский код иногда пишется на хинди и транслитерируется в C, C++ или Java. По некоторым сведениям, индусский код поют и танцуют, и только потом записывают на хинди. Есть народная примета: если программист поёт и танцует — значит, он написал индусский код.

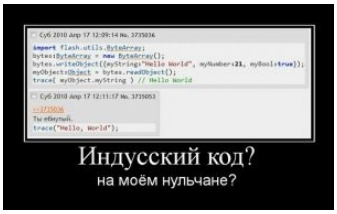


При частой работе с этими «спецами», фраза «The only good Indian is a dead Indian» обрастает новым смыслом.

[illegible], так-то!

Занимательная география

А на самом деле, китайцы ещё хуже, просто пришли в индустрию позднее, вот и проебали всю славу соседям по глобусу. Код, написанный индусами, обычно содержит в себе хоть какие-то зайчатки логики. Примеры выше это подтверждают (кто не понял, что там происходит, пусть первый бросит… курить, например). Код, написанный китайцами, парсится гораздо, гораздо сложнее. Переменная *сколькоМнеЛетЕслиКрокодилБолееШирокийЧемЗелёный*, не смотря на исчерпывающее название, считывается китайцем как . Поэтому он смело может дописать пару строчек кода, которые копируют в эту переменную («А мы проверяли, она с этого момента нигде не используется, мы же хотим сэкономить память?») младшие 17 битов от хеша кук браузера, и считать, что сделал доброе дело. Некоторые антропологи связывают такие особенности китайского разума с необходимостью заучивать 100500 иероглифов. Хули им после этого какие-то там магические константы или порядок смены смысла у каждой из тысяч переменных.



Но даже на фоне китайского, нихонский код заставляет вспоминать япону-мать. Если китайцы понимают своё скромное место в мире и всячески мимикрируют под сынов западной цивилизации (довольно часто понаехавшие из Поднебесной просят бледнолицых не ломать себе язык и называть их Джонами, Майклами и другими человеческими именами), то джапы обладают чудовищным комплексом полноценности. Их уже достаточно напрягает тот факт, что эти ваши джавы-сиприплюсы содержат прискорбно большие количества богомерзкой латынницы, опускаться до

https://www.youtube.co
v=jqLPHrCQr2I
Герои Intel

ингриша в комментариях для комфортного чтения лупоглазыми гайдзинами они не станут. То есть, там, где индус или китаец напишет:

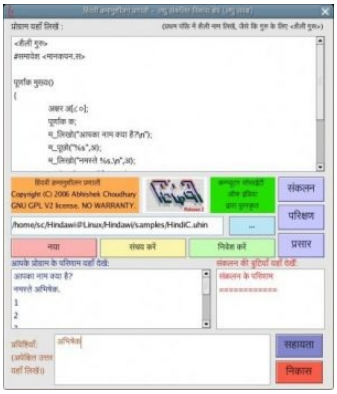
```
int randomInt = 4; // 4 is random enough
```

...и спалится сразу, японец снабдит эту строку кучей кракозяблов, причём файл может быть сохранён вместо Юникода в какой-нибудь локальной кодировке, про которую беложопые чукчи должны самостоятельно погуглить, дабы было что совать в онлайновый переводчик. Сколько иероглифов ты выучил сегодня?

Выше некоторые граждане хитрожопо пытались выдать следующий кусок кода за индусский:

```
#define true false
…
#define true (Math.random()>0.05) // Как вариант
```

Что есть, на самом деле, жуткий поклёп. «Карма» для индуса не пустой звук: он верит, что за такое западло Шива отхуярит ему ручки молнией, а Кали насрёт после этого в обугленную глазницу. Если индус и пишет хуёвый (с точки зрения другого программиста) код, то не по злобе, а по совершенно другим соображениям. А вот кто в реальности любит такие приколы, так это братья-славяне (в ходе марксистского воспитания лишённые поповских предрассудков), которые считают, что им несправедливо заплатили. Кроме того, оставлять такие закладки в коде свойственно людям с менталитетом школьников младших классов, независимо от страны произрастания, которым такие вещи кажутся верхом хитрости (поскольку с понятием w:Code review не знакомы).



Настоящий индусский код

Но ваистену страшнее всех зверей русский с двумя высшими образованиями, вооружённый книгами Александреску, ~~Чаушеску~~, Майерса и прочих парней, в жизни не написавших даже Doom'а. Такому дай стеклянный хуй — он из него сделает STL, а руки потом изрежешь себе ты, когда вместо *myArray.Add(1488);* будешь писать нечеловеческие заклинания типа *my_std_vector.push_back(1488);*.

Вывод? Нахуй расизм. Хороший код, независимо от национальности, пишут только два человека: ты да я, но я сейчас вместо кода пишу эту хуйню, а ты, вместо того, чтобы писать код, её читаешь.

Индусский дебаггинг

Частично является разновидностью индусского кодинга. Суть заключается в том, что при дебаггинге индус убирает видимое проявление проблемы, а не саму проблему (он её даже и не пытается искать). Например:

Проблема

При чтении одного конкретного ре́корда из базы прога падает в функции CalculateTaxes с ошибкой переполнения.

Человеческий фикс

```
  IF rs.fields("money") > 1e9 THEN
    Throw New System.Exception("Не может в нашей школе быть таких зарплат. В базе неправильное значение зарплаты для работника " + rs.fields("name") + ", исправьте.")
  END

  Call CalculateTaxes(rs)
```

Индусский фикс

```
  IF rs.fields("name") = "уборщица тётя люся" THEN
    ' Ну и похуй, что налоги для тёти люси не посчитаются. Зато софтина не упадёт!
  ELSE
    Call CalculateTaxes(rs)
  END
```

Примеры индусского кода

Пример № 1 (C#)

```
uint i;
...
if (i.ToString().Length == 1)
{
  ...
}
```

Не сразу можно понять, что в этом коде просто-напросто выполняется проверка `0 <=i <10`. Алгоритм достаточно прост: выполняется преобразование i в строку, после чего вычисляется её длина. Если число больше 9, то его десятичная запись содержит больше одного символа. Поскольку отрицательные числа переменная типа uint содержать не может, то проверку успешно проходят лишь числа от 0 до 9. Строка же, полученная из отрицательного числа, состоит как минимум из знака '-' и одной цифры, поэтому проверка справедлива и для знаковых целых (int). Алгоритм ресурсоёмок, неочевиден, и не поддаётся сопровождению даже теоретически.

Спискота говнопримеров

Пример № 2 (C#)

```
double rest(float a, float b)
{float res=a*b;
for (int i=0; i<999999999; i++)
 if (i<=res && i+1>res) {res = res-i; break;}
return res;
}
```

Далеко не сразу можно понять, что этот код выделяет дробную часть произведения a*b. Кроме того, функция выдаст ошибочное значение при a*b>999999999 или a*b<0. Также зависимость времени выполнения от величины целой части тоже не радует. К тому же, компилятор выдаст ошибку о несовпадении типов.

Пример № 3 (C#)

```
bool IsNumber (string str)
{
return (str.Replace ("0", "").Replace ("1", "").Replace ("2", "").Replace ("3", "").Replace ("4", "").Replace ("5", "").Replace ("6", "").Replace ("7", "").
Replace ("8", "").Replace ("9", "").Length == 0);
}
```

Хотя в данном случае алгоритм вполне очевиден, не менее очевидно и то, что для его выполнения конструктор класса string будет вызван не менее десяти раз (т.к. любое изменение класса String в C# приводит к созданию нового экземпляра и передачей старого экземпляра сборщику мусора) со всеми вытекающими из этого последствиями. И всё только потому, что какому-то индусу было лень придумать менее ресурсоёмкую альтернативу.

В случае множественных замен надо использовать StringBuilder.Replace(), а в данном - int.TryParse(). Не изобретайте велосипед! Кроме того, сам алгоритм содержит ошибку: он будет некорректно обрабатывать отрицательные числа. Добавление еще одного Replace("-","") к win не приведет, функция может вернуть труъ для строки "-".

Пример № 4 (C#)

```
bool value;

…

if (value.ToString().Length == 4)
{
  ...
}
else if (value.ToString().Length == 5)
{
  ...
}
else
{
   // внимание! кто-то хочет нас наебать подсунув некачественный (по всей вероятности, протухший) bool, но мы ему не дадим:
   throw new ArgumentException();
   // у некоторых племенных индусов бывает и продолжение, на случай если throw вдруг не сработает:
   return !true && !false;
   // надо же вернуть какую-нибудь хуйню, чтобы заказчик не ныл
}
```

В этом примере проверка истинности значения логической переменной производится с помощью длины её текстового представления. Если длина равна четырём („True“), значение истинно, если пять („False“) - ложно, если ни то, ни се - тогда включаем panic mode и начинаем жарить карри. Правда злобный компилятор вяло ругнется ворнингом. Но какой дурак их читает?

> Кроме привычной нам логики с двумя значениями (ложь и истина), существуют так же и другие логики. Примером может служить логика, основывающаяся на доказуемости теорем. Несложно заметить, что из не недоказуемости теоремы не следует ни её доказуемость, ни её недоказуемость (теорема может быть не разрешима в данной системе аксиом). Так что возможно в коде автора есть глубокий расчёт на будущее, но это вряд ли.

Алсо, кошерный вариант примера доставляет не меньше:

```
bool value;
…
switch (value)
{
  case true: ... break;
  case false: ... break;
  default: throw new ArgumentException();
}
```

Пример № 5 (PHP)

```
if ($_POST["end_oplata"]!="sending" and $_POST["continue_oplata"]!="prodoljit" and $_POST["prov"]!="proverka") {
...
}
elseif ($_POST["continue_oplata"]!="prodoljit" and $_POST["prov"]!="proverka") {
...
}
elseif ($_POST["continue_oplata"]=="prodoljit") {
...
}
```

В этом примере проверяется, какое действие нужно выполнить. Вместо того чтобы создать одну переменную со значением следующего действия, создается три «быдлокодерских» переменных.

К тому же, это пример ужасного стиля именования переменных, containing as English words, Tak i transliterirovannye Russkie. Видимо, сказался author’s poor английский словарный запас.

Пример № 6

На этой странице есть ссылки на страницу Индус. Если перейти на эту страницу, то Вы окажетесь опять же здесь. Пожалуйста, не правьте эти ссылки — они сделаны специально для примера.

Пример № 7

TurDuckEn Code — разновидность быдлокода, чаще всего встречающаяся в веб-приложениях. Как спагетти-код, только хуже. ТурДукен (en.w:Turducken) — пиндосская нямка, состоящая из индейки (turkey), нафаршированной уткой (duck), зафаршированной курицей (chicken). Турдукен Код, например, состоит из PHP, фаршированным SQL, нафаршированным HTML, вперемешку с CSS и зафаршированным Javascript. Вот так.

Пример № 8 (C++)

Вычисление интеграла

```
#include <math>

double f(double d){
return fabs(cos(d));
}

double integralrect(const double& a, const double& b, const double& epsilon)
{
double result;
int i;
int n;
double h;
double s1;
double s2;

n = 1;
h = b-a;
s2 = h*f((a+b)/2);
do
{
n = 2*n;
s1 = s2;
h = h/2;
s2 = 0;
i = 1;
do
{
s2 = s2+f(a+h/2+h*(i-1));
i = i+1;
}
while(i<=n);
s2 = s2*h;
}
while(fabs(s2-s1)>3*epsilon);
result = s2;
return result;
```

```
}
```

и, собственно, нахождение самого интеграла:

```
double s=integralrect(0,M_PI,0.0001);
s++;
```

Пример № 9 (Java)

```
public void write(){
     System.out.print("(");
     if (a.getHard()!=null){
      a.write();
     }
     else if (a.getConst()!=null){
      Const t=a.getConst();
      System.out.print(t);
     }
     else if (a.getVal()!=null) {
      Val t=a.getVal();
      System.out.print(t);
     }
     System.out.print(op.getOp());
     if (b.getHard()!=null){
      b.write();
     }
     else if (b.getConst()!=null){
      Const t=b.getConst();
      System.out.print(t);
     }
     else if (b.getVal()!=null) {
      Val t=b.getVal();
      System.out.print(t);
     }
     System.out.print(")");
    }
```

А нормальные люди пишут так:

```
public void write(){
     System.out.print("(");
     a.write();
     System.out.print(op.getOp());
     b.write();
     System.out.print(")");
}
```

А ещё более нормальные вот так:

```
public void write() {
    System.out.print("(" + a.getVal() + op.getOp() + b.getVal() + ")");
}
```

Пример № 10 (SQL)

```
SELECT CASE WHEN a>b THEN 1 ELSE NULL END
```

«Далеко не сразу можно понять», что ELSE является избыточным в данном контексте. Оператор CASE при отсутствии верных условий возвращает NULL.

Пример 11 (C/C++)

Настоящий индус добьется втрое больше случайности генератора чисел:

```
RANDOM = (rand() + rand() + rand()) / 3;
```

На самом деле, согласно центральной предельной теореме, итоговое распределение будет гораздо ближе к нормальному, чем к равномерному.

Пример 12 (PHP)

```
echo ("<form action='' method='POST'>");
echo ("<input type='text' name='username' /><br />");
echo ("<input type='password' name='password' /><br />");
echo ("<input type='submit' name='submit' value='submit' />");
echo ("</form>");

include "config.php";
$pass = (isset($_POST["password"])) ? $_POST["password"] : NULL;
$user = (isset($_POST["username"])) ? $_POST["username"] : NULL;

$password['0'] = (!empty($_POST["password"])) ? $_POST["password"] : 1;
$username['0'] = (!empty($_POST["username"])) ? $_POST["username"] : 1;

$password['1'] = (!empty($_POST["password"])) ? $_POST["password"] : 2;
$username['1'] = (!empty($_POST["username"])) ? $_POST["username"] : 2;

$query = mysql_query("SELECT * FROM admins WHERE password='".$pass."' AND username='".$user."'") or die(mysql_error());
$row = mysql_num_rows($query);

$query1 = mysql_query("SELECT level FROM admins WHERE username = '".$username."'");
$row1 = mysql_fetch_row($query1);

if (isset($_POST['submit'])) {
    if ($pass == NULL || $user == NULL) {
        die("ERROR");
    }
}
```

Nuff Said

Пример 13 (Python)

```
list = [0,1,2,3,4,5,6,7]
LisT = [0]
LisT = LisT * len(list)
var = list[0]
for i in range(len(list)):
    LisT[i] = list[len(list)-i-1]
LisT[len(list)-1] = var
print LisT
```

Для сравнения:

```
list = [0,1,2,3,4,5,6,7]
print list[::-1]
```

или даже

```
print range(7, -1, -1)
```

Пример 14 (1c)

Из реального коммерческого ПО, защищаемого железными ключами. Получение текущей даты:

```
Документ = СоздатьОбъект("Документ.ПКО");
Документ.Новый();
Документ.Записать();
ТекДата = Документ.ДатаДок;
Документ.Удалить(1);
```

Кроме того, что это «не совсем красиво», так еще и операции с документом приводят к манипуляции с записями в БД, что отнюдь не прибавляет скорости, да и надежности всей системы в целом. Особенно если учесть, что DBF версия записи фактически не удаляет, а просто помечает их внутри файла и тот пухнет, пока его не сжать спец утилитой.

А ведь можно немного проще:

```
ТекДата = ТекущаяДата();
```

Пример 15 (C#)

Также несколько байтоцентов аутсорцеры получают, дополнительно используя try ... catch:

```
private String GetCustomersList(out String error)
{
  try
  {/* check database */
    DataItem dataItem = (DataItem)m_appConfig_dataBase.GetDataItem("item_customerslist");
    if(dataItem == null) throw new NullReferenceException(); //4/11/2010--chupi21: no darta found
    return dataItem.ToString();
  }
  catch(NullReferenceException nullReferenceException)
  {
    error = "Data not found!";
    return nullReferenceException.ToString();
  }
}
```

Ентo еще не все! Так вот проверяют на ошибку методом:

```
String getCustomerListError = "Success";
String customerList = GetCustomersList(out getCustomerListError);//  get customer list
if(getCustomerListError == new NullReferenceException().ToString())
   return new ArrayList();
```

Да... вот так можно запутаться в собственном шедевре. Счастливчиком таки оказался наш мистер Чупи: список клиентов в базе таки создается при создании самой базы... Иначе SuxxAss бы он получил, а не Success.

Китайский код

Китайский код — стиль написания программ, нарушающий принцип DRY. Китайский подход к программированию требует эксплиситного (явного) отказа от циклов, локальных переменных, любых процедур и условных выражений, а также использования технологии copy-and-paste чуть менее, чем везде. Такой подход точно увеличивает объём исходников и может увеличить производительность (ведь пропускаются такты на джамповые команды)[1].

https://www.youtube.com/watch?v=m2cO0IEiiI4
Герои Intel, Чайна-стайл!

Возьмём, к примеру, такой кусочек программы на C:

```
int arr[10];
int i;
for (i = 0; i < 10; i++)
{
  arr[i] = 0;
}
```

Который, кстати, вполне мог бы выглядеть и так:

```
int arr[10] = {0};
```

Типичный программист в китайском стиле напишет это так:

```
int a0 = 0;
int a1 = 0;
int a2 = 0;
int a3 = 0;
int a4 = 0;
int a5 = 0;
int a6 = 0;
int a7 = 0;
int a8 = 0;
int a9 = 0;
```

...и в дальнейшем будет использовать a0, a1, a2, a3, a4 и т.д. Например, вместо прекрасного:

```
if (x < 10) arr[x] = x;
```

будет:

```
if (x == 0)
{
  a0 = x;
}
else if (x == 1)
{
  a1 = x;
}
else if (x == 2)
{
  ...
}
```

Пример № 1, приведённый выше:

```
uint i;
...
if (i.ToString().Length == 1)
{
  ...
}
```

...приверженец китайской методы перепишет так:

```
if (i == 0 || i == 1 || i == 2 || i == 3 || i == 4 || i == 5 || i == 6 || i == 7 || i == 8 || i == 9)
{
  // произвести ещё одну бессмысленную операцию
}
```

В то время как суть индусского метода заключается в как можно более полном затуманивании предназначения программного продукта, китайский код зачастую поражает и даже отупляет простотой и брутальной прямолинейностью подхода, что характерно для китайской инженерии в целом. С другой стороны, нельзя не отметить, что подобные технологии позволяют максимальным образом трудоустроить население и приводят к сокрушительным победам в социалистических соревнованиях по количеству написанных строчек.

Если программист в китайском стиле напишет процедуру, то вероятность того, что результат её деятельности будет совершенно бесполезным, стремится к единице.

Очень часто китайский код пишут тупые студенты, которые пришли учиться программировать по неясным причинам. Потом такой код приходится долго вкуривать. Выражение «ебу и патчу» произошло благодаря как раз таким случаям.

Делфишколокод

Явление, наблюдаемое исключительно на просторах Этой Страны. Возникло из-за маниакального пристрастия «преподавателей» информатики Этой Страны преподносить неокрепшим школьным умам «основы программирования» на базе небезызвестной быдлокодерской RAD Delphi. Особенностями такого кода являются стандартные автоматически сгенерированные RAD имена методов классов, событий, переменных (**Form1**, **Button1**, **Button1Click** и т. д.), отсутствие всякого форматирования, комментариев, документации. В случае, если делфишколокодер придумывает собственные названия методам, функциям или переменным — эти названия состоят из транслитерированных русских слов вперемешку с немногими известными шололо английскими, например **CheckOplata**. Либо — вообще лишены всякого смысла. Примеры делфишколокода, а также примеры замечательных и очень полезных «программ», создаваемых нашими школьниками, можно найти на замечательном, зашкаливающем лулзами «Личном сайта Разработчика!» по адресу http://ykolchurin.narod.ru/. Лулзы от этого кода искать здесь: http://www.sql.ru/foru...d=708039

Оверинжиниринг

Кроме быдлокода и китайского кода, высокой пертинентностью к индусскому коду обладает т. н. «оверинжиниринг» — решение простых задач сложными методами. Да-да, просто чтобы повыёбываться, показать своё знание всех возможных приёмов, методов и конструкций (в чем и отличие от индусского кода) и произвести впечатление на человека, от темы далёкого, но зато платящего за это денежки. Как всегда, ничто не ново под луной: в 1920-е был крайне популярен карикатурный изобретатель с характерно громоздким именем «профессор Люцифер Горгонзолла Баттс», пародирующий первый виток явления, окружённый своими адскими механическими машинами, которыми познее вдохновились создатели досовской игрушки «The Incredible Machine». Сейчас то же самое наблюдается как в электронике, так и (особенно ярко) в сфере ПО. Причём опенсорс подвержен этому раку не менее, чем коммерческий код: там тоже пытаются произвести впечатление на пустом месте, но уже не на заказчика, а на тусовку. Такие дела.

Большой недоговоркой было бы умолчать, что сие явление не просто масштабно — оно всеохватно, о́бло, озо́рно, огро́мно, стозе́вно и ла́яй ©. Раком оверинжиниринга отрасль охвачена, пожалуй, полнее, чем рынок — быдлодевайсами, а телевизор — зомбопрограммами. Она охвачена им практически *вся*. Результаты, собственно, печальны и наблюдаемы невооружённым глазом. Умирающий от ожирения код ~~пишется~~ визуально генерируется умирающими от ожирения средствами разработки, а количество багов в результате просто астрономическое (преподносится этот бред как «средства, помогающие минимизировать людские ошибки» — быдломенеджеры по внедрению верят буклетам больше, чем визуально наблюдаемой картине). Скорость достижения результата тоже прямо противоположна заявам (на ассемблере и то было бы быстрее написать). Вопрос «нахуя?» вызывает в ответ тонны шизофазии про «прогрессивные методы и смелый взгляд в будущее». Запасаемся попкорном и ждём прорыва этого гнойника.

Пример кода: Hello world на PHP *in da patterns*.

Анналы истории

29 января 2009 года, на открытии Всемирного экономического форума в Давосе, афтаритетность и квалифекация наших индусско-индийских коллег была подтверждена на официальном уровне и использована для доказательства не меньшей авторитетности и квалификации нашего брата:

> «У нас традиционно хорошо развита математическая школа, и программисты у нас одни из лучших в мире. Это без всяких сомнений. Думаю, что с этим никто не будет спорить, даже наши индийские коллеги. »
>
> — *[1]*

Олсо, 26 ноября 2009 года Александр Лукашенко выступил перед участниками специального заседания саммита «Соединим пространство СНГ» с почти дословным пересказом своего бывшего российского коллеги:

> «В Беларуси традиционно хорошо развита математическая школа, наши программисты - одни из лучших в мире. Поэтому нам есть что предъявить в качестве интеллектуального продукта »
>
> — *http://www.president.gov.by/press80066.html*

Следует отметить, что подобное сравнение проводил также Билл Гейтс, который во время своего визита в Россию высоко оценил возможности России и потенциал российских специалистов в области высоких технологий и точных наук, и, соответственно, во время последнего визита в Индию, их индусских коллег.

Последствия

Ахтунг! Не всё из нижеприведённого на совести индийских расовых индусов, поскольку «индусский код» давно имя нарицательное.

Подводная лодка «Нерпа»

Российская подлодка, построенная для передачи в дружественную Индию, на пути к которой в 2008 году случайно система пожаротушения, в результате чего к Кришне отправились 20 человек экипажа.

«Мичман также считает, что произошел сбой в компьютере. «Такую систему нельзя было запитывать на электронику. Раньше на всех лодках система пожаротушения была ручная, по приказанию. Дается команда: „Включиться в дыхательные аппараты!“, и только потом запускается система тушения. А тут уже фреон льется и одновременно аварийная тревога. У нас был пульт общекорабельных систем „Молибден-И“, **индийский вариант**», — сказал Кошеваров.

По его словам, экипаж во время аварии действовал грамотно. Многих из примерно 70 человек, находившихся во втором отсеке, спасли.

»

— *http://www.newsru.com/russia/12nov2008/podlodka.html*

Кстати, кодовое имя пепелаца — *Chakra* («Чакра»), что несомненно символизирует.

Therac-25

Терак-25 (англ. Therac-25) — аппарат лучевой терапии, медицинский ускоритель созданный канадской государственной организацией Atomic Energy of Canada Limited. От него погибло как минимум шесть человек. А все это из-за сами знаете чего.

Обезьяны, пейсавшие код были довольно везучими и допустили около 3 ошибок. Среди них шедевры:

- Одна и та же переменная применялась как для анализа введённых чисел, так и для определения положения поворотного круга. Поэтому при быстром вводе Therac мог иметь дело с неправильным положением поворотного круга.
- Настройка положения отклоняющих магнитов занимает около 8 секунд. Если за это время параметры типа и мощности излучения были изменены, и курсор установлен на финальную позицию, система не обнаруживала изменений.
- Установка булевской переменной (однобайтовой) в значение «истина» производилось командой «x=x+1». Поэтому с вероятностью 1/256 (если x=255, то сложение с единицей приводит к переполнению и обнулению переменной) при нажатии кнопки «Set» программа могла не пропустить информацию о некорректном положении диска.

Ariane 5

История о том, как можно написать **одну** строчку кода ценой в девять нулей после единицы. Долларов.

«

```
P_M_DERIVE(T_ALG.E_BH) := UC_16S_EN_16NS (TDB.T_ENTIER_16S
                                         ((1.0/C_M_LSB_BH) *
                                         G_M_INFO_DERIVE(T_ALG.E_BH)))
```

»

— *Виновница торжества*

4 июня 1996 года в 9 часов утра ~~вероломно, без объявления войны~~ конвертация 64-битного вещественного числа в 16-битное знаковое целое закончилась арифметическим переполнением. Не вынеся такого позора, ракета *Ariane 5*, на чьём бортовом компьютере произошло сие безобразие, по-самурайски самовыпилилась на 40-й секунде полёта, прихватив с собой товарно-материальных ценностей (4 спутника ЕКА «Cluster», предназначенных для изучения магнитного поля Земли) на $350—500 млн. С учётом всех потерь (отложенные ~~кирпичи~~ запуски, подмоченная репутация и т.п.) общая сумма перевалила за гигабакс, сделав эту ошибку возможно самой дорогой в истории.

Благодаря чему стало возможным это эпическое свершение?

- Соответствующий код был в лучших традициях индуизма копипастой с предыдущей ракеты *Ariane 4*. Копипакостники, ЧСХ, не удосужились проверить, работает ли он в новом окружении.
- Потрясающая обработка исключительных ситуаций: при возникновении в аппарате странных результатов, считаем происходящее аппаратной ошибкой с отключением аппарата нахрен. Один из экспертов, проводивших вскрытие, сравнил такой механизм с врачом, без всякого осмотра пристрелившим пришедшего к нему с непонятными симптомами больного, дабы тот не мучился. Сюрприз-сюрприз, этот подход жив и поныне, и изредка являет миру свой лик в виде BSOD. Разница в том, что пекарня анона в таких ситуациях просто зависает, а управляющая система *Ariane 5* после этого передала управление точно такому же компьютеру (!) с точно такой же программой (!!). Так как баг на самом деле был отнюдь не аппаратный, а самый что ни на есть программный, конец немного предсказуем, не так ли?
- Комиссия, проводившая расследование, копнула глубже, и пришла к выводу, что за предыдущими пунктами скрывается принцип: «Программа считается работающей верно, пока не доказано обратное». В то время как должен применяться принцип: «Программное обеспечение нужно считать ошибочным, пока использование практических методов, признанных в настоящее время наилучшими, не докажет его правильность».

Коммерческие продукты, содержащие индусский код

- Microsoft Windows. Исходники NT4, 2k, xp есть в сети. Найти в сорцах следы индусов не сложно. В 2014 M$ окончательно перестала притворяться, взяв *генеральным директором* индуса!
- Symbian. Горячо любимая нокией. До открытия исходного кода писалась индусами. Я гарантирую это. Те, кому нужен пруф, могут посмотреть на обилие индийских знатоков symbian на форуме http://discussion.forum.nokia.com/forum/
- Драйвера AMD/ATI. Эдакий эталон говнокода из парижской палаты мер и весов. Количество багов, глюков и недоделок перевалило за все мыслимые пределы (особенно на Линуксе). Радует также Catalyst Control Center, написанный на б-гомерзком Microsoft.NET. Впрочем, это не удивительно — сама контора анально зависит от мелкомягких. Алсо, ходили слухи, что последние версии драйверов видеокарт писали их ближайшие конкуренты — калифорнийская хлебопекарня nVidia.
- Тысячи их.

Команда разработчиков IE7.
Вглядитесь в эти лица.
И на количество пальцев, что показывает нижний-правый программист

См. также

- Копипаста:Быдлокод-ГСР
- Быдлокодер
- Программист
- Админ
- Индийское кино
- Yandere Simulator

Примечания

1. ↑ То, что это шутка, должно быть очевидно. А вот чего ты мог и на самом деле не знать, так это того, что в Майкрософте одно время все продукты для пользователей собирались с ключом O1 — «оптимизация

исполняемого файла по размеру» (после чего бинарник начинает состоять из джамповых команд чуть менее, чем полностью). Прикол в том, что субъективно он может восприниматься как более быстрая программа, ибо тупо меньше читать с диска и писать в память плюс меньше нагрузка на своп).

IDA Pro срывает покровы

Ссылки

- Блог о загадочных индийских техниках программирования.
- ЖЖ-сообщество programmers_fun.
- ЖЖ-сообщество code_wtf.
- КодоБред на Хабре.
- Русский код, бессмысленный и беспощадный.
- The Daily WTH (бывшая The Daily WTF) (бывшая Worse than Failure (бывшая The Daily WTF)).
- Пример индусского кода на БОРе.
- И ещё.
- Удаление аппендикса через рот.
- Говнокодеры едут на работу по рабочей визе H1B.

Языки программирования

++i + ++i 1C AJAX BrainFuck C Sharp C++ Dummy mode Erlang Forth FUBAR God is real, unless explicitly declared as integer GOTO Haskell Ifconfig Java JavaScript LISP My other car Oracle Pascal Perl PHP Prolog Pure C Python RegExp Reverse Engineering Ruby SAP SICP Tcl TeX Xyzzy Анти-паттерн Ассемблер Быдлокодер Выстрелить себе в ногу Грязный хак Дискета ЕГГОГ Индусский код Инжалид дежице Капча КОИ-8 Костыль Лог Метод научного тыка Очередь Помолясь Проблема 2000 Программист Процент эс Рекурсия Свистелки и перделки Спортивное программирование СУБД Тестировщик Умение разбираться в чужом коде Фаза Луны Фортран Хакер Языки программирования

Еда

11 сентября Ache666 Alt-Right Avatar Beon.ru Boku no Pico Butthurt Championat.com Check you Chris-chan Cruel Addict Du Volon Fandom He Will Not Divide Us Leyla 22 Limp Bizkit Lingqiyan Linkin Park Megadeth MISS HOLLYWOOD Noize MC Project N.I.G.R.A. Pussy Riot Ray William Johnson Rsdn.ru Rutracker.org SupLisEr VIP X не умер Yandere Simulator Zeitgeist Аббатус Авраам Болеслав Покой Адольфыч АлисА Алкснис Аллан999 Альбац Альберт Акчурин Андерс Брейвик Андрей Лаптев Андрей Сковородников Анна Бешнова Апач Бабка АТС Багиров Бачинский Белоцерковская Белый Колонизатор Бобби Котик Бодибилдинг Болашенко Бурление говн Валерий Назаров Варракс Леонид Василевский Ватник Веганы Миша Вербицкий Винилофилия Виталик Вован Метал Высер Геноцид армян Германыч Глобальное потепление Гоблин Гага Говнарь Говно Город Снов Гринпис Гутник Дед ИВЦ Джеттейм Джигурда Джипсилиля Диванные войска Доброволец Друмба Дэниел Петрик Евровидение Еда Женя Духовникова Жертвы пранка Закон По Змагар Знаменитость российского уровня Иван Гамаз Иван Охлобыстин ИГИЛ Илья Фарафонов Император двачей Индусский код Инна Жиркова Кавказ-Центр Кактус Карикатуры на Мухаммеда Карина Будучьян Кинопоиск Коммуняки