####?

######

########

2009 ####### ## ### ######### ########## #######–#### ##### ####://###.######.###/########–##########–###########–#####–###/.

monad ####### #### ## ######### ## ## ######## #### ### ########## ## ######### ## ### #####–### ####### ## ###### ### #### ## ## ########### ######### ## #### ##### #### ## ########## ####–########. ### ####### #### ## ######### ## # ### #### ### ######### ## ######### ###### #####–######## ## ##### ### ######### ############ ### #### ### ####### ### ########## ########### ## ######## ######.

#########, ################ ### #########, ##### #### #### ## #########.

######### ### ######## ###### ## ######### ## #### ###### [######:############.###] ######## #, *Tony Morris - PGP Key*.

########

Introduction

[###########1988]. ###### ###### ### ##### ######–##### #### ###### ####### #### ### ####### ## ########### #### ## ###### ########## ########### #########.

###########. #### ## ### ####. ##–####, ### #### ########### ########### ## ###### ##### # ########. #### ###### ######### ########### ### ### ###### ## ########### ### ########### (### [#####]). ###### ### #### ## ##### ###### ########### ########. ### ##### ########### ## ####### ## ### ### #### ## ##### ## ### ####.

##########, ### #### ### ######## ############. ## ### ########## ## ##### ############ ### ###, ### #### ## ######## ## #### ######### ## #### ####### ## #### #####–### ########### ### ####### ##### ############ #######.

Mythology

Debunking

######### ####### ##### ## ##### ####### #### ###########, ####### ### ########, ## ##–#### ## ## ####, ##### #### ### #### # #### #### ## ### ## ####### ### #############.

> ###### ######## ###### ## ####### ### #### ####### #### ######. ####### #### #### ######## #### ## # ###### ## #######.

####, #### ##### ### ######### ######## ####### ### ####### #########. ## ### ###### #####, ##### ####### ### ###### ########, #####, ####### #########. # ### ## ##### ######.

Monads are a hack for working with I/O in the real world

Other forms

- ###### ### ### ########### ########### ######### #### ### ## ## #########.
- ###### ### ### ########### ####–#######.
- ####### ###### ## ######### ## ###### ## #/#.

Demystify

####### # ######## #### ######## # `List` ### # ##### ###### `Banana` #### #### ##### ##### ### ### ## ### ######.

```
<A> List<A> reverse(List<A> list)
class Banana {}
```

#######, ### ## ## ######### ## ### #### ### `reverse` ######## ## ## #### ### ####### ## #######. ######### # #### ## ####### ## ###### # ###### ######## ## ## ######## ###########. ### ######### #### ######, ## ## #### ######### ## ### #### ###### ### ## #### ### ####### ## #/#.

##########, ### ## ## ######## #### (####) ####### ## ### #### ###.

Monads are only for functional or esoteric programming languages

Other forms

- # ##### ### ###### ## ## #### (## .### ## #### ## ###### ## ###### ## ...) ##########.
- ###### ### ### ###### ##### ##### #########, ### ## ### ## ########.

Demystify

- ####: ### ###########, ######### ## ###, ### ###### ## ##### 100 ##### ### ###. #### ### # ##### ## ##.
- #### #### #####, ## ## ###### ###### #### ### ######### ### ###### ## ## # #### #### ##–########### ## ### ##### ####### ### ## ######### ### #### ###### ## ##########.
- ###### ### ## ##### ## ###### ## ####–###### ## ######## ######## #### ## ####.

Monads make my program impure

#########. #############, #### ### ######## ## ######## ######### ## (##### ## ### ######, ### ########### ####### #### #####) ##### ######## ####, #### ##### ######### ## ######.

Terminology

A quick note on terminology

####, #### ############, ##### ### ######–##### ############.

Monad laws

What do we mean by the *Monad Laws*?

- ### ### #### ##### ## ### ##### ####. ##### ### ##### ## ####. #### #### #### #####![1]

[1] ############, #### ########, ##### ########

- #### ## ########### #### #### #### ### ### ######## ## ### ########.
- ### ### ### ######### #### #### ## ####.

For example

####### 1. ## ############ **`List`** ########## ##### ####

```
interface List<T> {
  T get(int index);
  int length();
  List<T> reverse();
}
```

- ## ### ## #### #### ### ######## (`list`) #### ####### `list.get(list.length() - 1) == list.reverse().get(0)`
- ##### ### ##### #### #### ### ######### ## #### ########### #########.
- ### ##### ######### #### ######## ##### ####, ### ## ##### ## #### ####. #######, #### ## ##### ## #### ## #### ## ### ##### ####.

Type constructors and kinds

- ## #### 1.5, `List` ## ### # ####, ####### ## ######## # #### ######## [2].
- `List<Long>` ## # ####, ### `List` ## # *type constructor*.
- `List` ### # *kind* ##### ## ######## `* -> *` ### ##### ###### ### #### ## ###### # #####.
- `HashMap` ## # #### ########### ##### ## ###### `(*,*) -> *` ##### ## ##### ### #### ######### ## ###### # ####.
- `List<Long>` ### # *kind* ##### ## ######## `*` (#.#. ## ## # ####).

Higher-order Polymorphism

- ######### #### ## #### ### ## #### *first-order polymorphism* ####### #### ##### ## ## ##–###### ## #####. #.#. `List<A>` ### #### # `reverse` ######## #### ##### ## ### ####### #### (### `A`).
- #### ######### ########### ######### ### #### #######[3] ##### ## ## ######## ## *type constructors* ## ####.
- #### ####### ## ###### ######–##### (## ######–######) ############.

[2] ######## ####### ############ ##### `List` ## ########## ## `List<?>`
[3] #######, #####, ##### (### ##)

So what is a Monad then?

It's a fluffy cloud

#####

Produced by a sonic boom

########

Monad metaphors are stupid

##, ## ##### ######### ### ###### ### #########.

######### ### ######## ###### #### ####### ######## ############# ### ##### ######## (######–######## ########### ######?).

##.

First let's pretend that Java/C# are practical

####### 2. ######–#### #### ## ######## ######## ### ######–##### ############[4]

```
interface Transformer<X, Y> {
  Y transform(X x);
}

interface Monad<M> { // M :: * -> *
  <A> M<A> pure(A a);
  <A, B> M<B> bind(Transformer<A, M<B>> t, M<A> a);
}
```

Finished staring?

#########[5]. #### ## ### ## ##.

####### 3. ### #### #####

```
new Monad<List>() { // List :: * -> * (kind checks)
  public <A> List<A> pure(A a) {
    return List.single(a);
  }

  public <A, B> List<B> bind(Transformer<A, List<B>> t, List<A> a) {
    List<B> r = List.empty();
    for(e : a) r.addAll(t.transform(e));
    return r;
  }
};
```

Monads and more monads

####, ##### ### #### ## ######## ######### ## ### `Monad` #########. #### `Transformer` #### ### #### ######## ####### ## # #####.

[4] ## ## #### # ###### ###### ######

[5] ### #### ######### ### ##### ##### ####

####### 4. ### ###########<#, _> #####

```
new Monad<Transformer<X, _>>() {
  public <A> Transformer<X, A> pure(A a) {
    return new Transformer<X, A>() {
      public A transform(X x) { return a; }
    };
  }

  public <A, B> Transformer<X, B> bind(
        Transformer<A, Transformer<X, B>> t, Transformer<X, A> a) {
    return new Transformer<X, B>() {
      public B transform(X x) { return
        t.transform(a.transform(x)).transform(x); }
    };
  }
};
```

More examples

####### 5. # ##### ## # #####[6]

```
interface Value<T> {
  T value();
}

new Monad<Value>() {
  public <A> Value<A> pure(A a) {
    return new Value<A>() { public A value() { return a; } };
  }

  public <A, B> Value<B> bind(Transformer<A, Value<B>> t, Value<A> a) {
    return t.transform(a.value());
  }
};
```

Other examples include

- ######## ######
- ######## ##########
- ######## #####
- #############
- #########
- #####
- #######
- ####–####### (!!!)

[6] #### ######### ###### ### ######## #####

So what, why do I care?

Repetition is for solving non-problems using clumsy languages

########. ####### ## # `sort` ######## #### ####### ### ##### #### ### ######## ###### ### ####### ## #######, ##### ######### ### ### ## ### #### ########### #### ### ######## ###### # #####.

- <#> #<####<#>> ########(####<#<#>> ####)
- <#> #<####<#>> #########(### #, #<#> #)
- <#> #<#> ####(#<#<#>> #)
- *etc. etc.*

And it's not just monads

higher-order polymorphism. ## ### #### ########## #######–#### ########, ########, ######, ######### ### ############# ########, ###### ######### ########, ######### ######### ### #### ##### ######## #### ### #### ########## ## ### ##########.

#######!

C#, LINQ Query Comprehensions and Monads

C# 3.0

- ######### ########## #### ##### ############## #### ## ####### 3.0.
- #### ###### ### ######## ## ######### #### ## ##### ########## #### ########## ## ### #######.
- ##### ### #### ##### #########! #### ##### ############## ### ######### ##### ### ####### #### *any* #####.
- #### ## #### ###### `bind`, ##/#### ##### `SelectMany` ### #### ## #### ###### `Transformer`, ## ##### `Func`. ## #### ########## ######### ##### ### ############# `Func`.

Syntax sugar

`x => method(x)` ## ###### ##### ### # `Func` ######## #### ##### # ######## (#) ### #### `method` ## ##.

####### 6. #### ##### #############[7]

```
from i in a
from j in b
from k in c
select foo(i, j, k);
```

[7] `from`, `in`, ### `select` ### ########.

####### 7. ####### ### ###### #####

```
a.SelectMany(i =>
b.SelectMany(j =>
c.Select(k =>
foo(i, j, k))));
```

LINQ

########, ### ##### ############# ### #### ## *any monad*. ##### ####### ##### ##### ###### ## ############ ### `SelectMany` (### `Select`) ######.

######–##### ############. ############, ## ###### ##### ######### ## ### ## ### #####.

(###### ####), ### ##### ### ## ##### #### ### ## ##!

Scala for-comprehensions and Monads

Scala

- ##### ### ### ###–############## #### ### #########, ### ### ###### ######–###### ############[8].
- #### ## ##### `SelectMany` ### ## #### ###### `bind`, ##### ##### `flatMap`.
- #### ####, ###–############## ####### ######### ##### ### ##### ## `flatMap` (### ###–###) ### #### #### #### ### ### ####–####[9].

For-comprehensions

####### 8. ##### ###–#############[10]

```
for(i <- a;
    j <- b;
    k <- c)
yield foo(i, j, k)
```

####### 9. ####### ###### #####

```
a flatMap (i =>
b flatMap (j =>
c map (k =>
foo(i, j, k))))
```

Transforming side-effects using monads

Suppose a program in a typical language

[8] ##### ## ####### (##### ####### 28, 2009) ##### ########### ###### #### ####### #### ###### ## ### #### ## ####### ### ##############.

[9] #### ## ####### ## ########## #######, ###### ##/####, ##### ## ### ## #### ###### ###########.

[10] `for` ### `yield` ### ########.

```
T t = e1();
e2(t);
U u = e3(t);
V v = e4(t, u);
return e5(u, v);
```

#####.

Change the syntax

- ###### ####–###########.
- #### ### ####/##### ## ## ##########.
- `void` ####### #### ###### # ##### #### ## ####### (##### #### ## _).

```
e1() = t;
e2(t) = _;
e3(t) = u;
e4(t, u) = v;
return e5(u, v);
```

More changes

- ####### ####–###### #### =>.
- ####### `return` #### `pure`.
- ####### = #### `flatMap`.

####### 10. ####!

```
e1() flatMap t =>
e2(t) flatMap _ =>
e3(t) flatMap u =>
e4(t, u) flatMap v =>
pure e5(u, v);
```

More fun

- ########### ######## [####://###.###.####.##.##/#####/######/###########.###]
- ###### [####://###.##.########.##/#####/###–######.###]
- ######### ######### (#########) [####://###.######.##.##.##/######.#######/#######–#####/########.###]
- #########?

Special Thanks

- ##. ###### #####, ########### #######, ######.
- ##. ####### #######, ####### ############, ########.
- ##. ###### ######, ###########, ######.

- ### ######### ############ ## ####### ####### ### ######## ### #### ######## #####–##### ###### ### ########## ## ########. #### ############ ### ###########.

############

[##########1988] ####### ##### . 1988 . *Computational lambda-calculus and monads* .

[#####] #### ###### . 1984 . *Why Functional Programming Matters* .

A. Code Examples

Scala

#####.#####

```
// Monad interface
trait Monad[M[_]] {
  def pure[A](a: A): M[A]
  def bind[A, B](a: M[A], f: A => M[B]): M[B]
}

object Monad {
  // Monad instances
  val ValueMonad = new Monad[Function0] {
    def pure[A](a: A) = () => a
    def bind[A, B](a: Function0[A], f: A => Function0[B]) =
      f(a.apply)
  }

  val ListMonad = new Monad[List] {
    def pure[A](a: A) = List(a)
    def bind[A, B](a: List[A], f: A => List[B]) =
      a flatMap f
  }

  val OptionMonad = new Monad[Option] {
    def pure[A](a: A) = Some(a)
    def bind[A, B](a: Option[A], f: A => Option[B]) =
      a flatMap f
  }

  trait PA[T[_, _], A] {
    type Apply[B] = T[A, B]
  }

  def Function1Monad[X] = new Monad[PA[Function1, X]#Apply] {
    def pure[A](a: A) = _ => a
    def bind[A, B](a: X => A, f: A => X => B) =
      x => f(a(x))(x)
  }

  // Monad functions
  def sequence[M[_], A](as: List[M[A]], m: Monad[M]) =
    as.foldRight[M[List[A]]](m pure Nil)((h, t) =>
      m.bind(h, (a: A) => m.bind(t, (as: List[A]) => m pure (a :: as))))
```

```scala
  def join[M[_], A](a: M[M[A]], m: Monad[M]) = m.bind(a, (z: M[A]) => z)

  def replicate[M[_], A](n: Int, a: M[A], m: Monad[M]) =
    sequence(List.fill(n)(a), m)

  // etc. etc. (Monad functions)
}

// Monad demo
object Main {
  import Monad._

  def main(args: Array[String]) {
    println(sequence(List(List(1, 2, 3), List(4, 5, 6)), ListMonad))
    println(sequence(List(Some(7), Some(8)), OptionMonad))
    println(join(List(List(1, 2, 3), List(4, 5, 6)), ListMonad))
    println(join(Some(Some(7)), OptionMonad))
    println(replicate(3, "abc".toList, ListMonad) map (_.mkString))
    println(replicate(4, Some(8), OptionMonad))
  }
}

// scala-2.8.0.r19410-b20091106023416

/*
List(List(1, 4), List(1, 5), List(1, 6), List(2, 4), List(2, 5),
  List(2, 6), List(3, 4), List(3, 5), List(3, 6))
Some(List(7, 8))
List(1, 2, 3, 4, 5, 6)
Some(7)
List(aaa, aab, aac, aba, abb, abc, aca, acb, acc, baa,
  bab, bac, bba, bbb, bbc, bca, bcb, bcc, caa, cab, cac,
  cba, cbb, cbc, cca, ccb, ccc)
Some(List(8, 8, 8, 8))
*/
```

C#

#####.##

```csharp
using System;
using System.Collections.Generic;
using System.Linq;

static class MonadFunctions {
  public static IEnumerable<A> joinEnumerable<A>(IEnumerable<IEnumerable<A>> a)
    // Forced repetition.
    return a.SelectMany(z => z);
  }

  public static IQueryable<A> joinQueryable<A>(IQueryable<IQueryable<A>> a) {
    // Repeats for each type constructor.
    return a.SelectMany(z => z);
  }

  // join*
}
```

```
class M {
  public static void Main(string[] args) {
    var i = new[]{1, 2, 3};
    var j = new[]{4, 5, 6};
    Print(MonadFunctions.joinEnumerable(new[]{i as IEnumerable<int>, j}));
  }

  static void Print<A>(IEnumerable<A> a) {
    foreach(A i in a)
      Console.Write("{0} ", i);
    Console.WriteLine();
  }
}
```

Haskell

#####.##

```
-- Monad interface
class Monad' m where
  bind :: m a -> (a -> m b) -> m b
  pure :: a -> m a

newtype Value a = Value {
  val :: a
}

-- Monad instances
instance Monad' Value where
  bind (Value a) f = f a
  pure = Value

instance Monad' [] where -- aka List
  bind = (>>=)
  pure = return

instance Monad' Maybe where -- aka Option
  bind = (>>=)
  pure = return

instance Monad' ((->) t) where -- aka Function1
  bind a f = \x -> f (a x) x
  pure = const

-- Monad functions
sequence' :: (Monad' m) => [m a] -> m [a]
sequence' = foldr (\h t -> bind h (bind t . (pure .) . (:))) (pure [])

join' :: (Monad' m) => m (m a) -> m a
join' a = bind a id

replicate' :: (Monad m) => Int -> m a -> m [a]
replicate' n a = sequence (replicate n a)

-- etc. etc. Monad functions

main = do print (sequence' [[1, 2, 3], [4, 5, 6]])
```

```
          print (sequence' [Just 7, Just 8])
          print (join' [[1, 2, 3], [4, 5, 6]])
          print (join' (Just (Just 7)))
          print (replicate' 3 "abc")
          print (replicate' 4 (Just 8))

{-
[[1,4],[1,5],[1,6],[2,4],[2,5],[2,6],[3,4],[3,5],[3,6]]
Just [7,8]
[1,2,3,4,5,6]
Just 7
["aaa","aab","aac","aba","abb","abc","aca","acb","acc","baa",
  "bab","bac","bba","bbb","bbc","bca","bcb","bcc","caa","cab","cac",
  "cba","cbb","cbc","cca","ccb","ccc"]
Just [8,8,8,8]
-}
```

B. Tony Morris - PGP Key

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v1.4.6 (GNU/Linux)

mQGiBETNyC0RBAC3MYSZSbDZhBLKra2YUphB9OO6+qMFl/v2Lq8590ZfeE2WjIOu
c/KGKyOigXztMrA4+iekUjM4FA8E6AlBRQiAqZK8HF0ftX5hDpuSyEKkZe3jcxxI
BbhwX/SWHtDEVzwNmvO1wEnwhRE1oY/BCmy+bQ9wmAjlNav4UbOIcXcJUwCgz6FF
UwDvPxzybgd9FNS14BE37n8D/AzGmGjunBW/x+g/ndwu6WEpTEn1ZNdja6VrNXSG
xQ8XM+NBulroAIYX+YWdHsKnuGvSKCgVoc1ifVHdztA9sksID5GBGzhmVbIP2E16
w/LEzqqwruv9dX0Y1b7n8hcnvbl4DgEgLgeQ+VgJfLUkY2jFZ2m3dEBRH9USSgB/
V2pBBACP4Us2ZBYEXtMG29g6GqyeeJLEP34PLYVHWJZbet/wPQsHFhYahhzjwZGG
Srp0JUpegJOdaqX/Y0nio2whgCpqJcrbGuUBqNgQI3k4gvBwnPyx7jM0kfHfFORG
lq9SDbfLpV0EJx6fWXGStW33zsQQvenDcV2F5czzKQcy66BFwbQhVG9ueSBNb3Jy
aXMgPHRtb3JyaXNAdG1vcnJpcy5uZXQ+iGMEExECACMFAkTNyC0FCQlmAYAGCwkI
BwMCBBUCCAMEFgIDAQIeAQIXgAAKCRCaemCth7qvrf0pAKCFii2pI2W1BKVFuQcw
yoNxP0CAUACgyhuI9isCvtrOkeyjDmCVueRCdaC5Ag0ERM3IThAIAJ6A1z+d40ve
WvPIhpFiGfoS8UX4YdgYpo2mC/orY0xszBitogtaTHQHU5YDemGg81plNg9I3DbM
Er8uyONV4DqF6RbLj4w+iA6zn93+PTEZ73ydhxF6vDuojpVZPXVzXzpgyXHkEVLC
3hKL9oVlEsh+DWCvCiSAIy780JZ3FNVuMC3VH4qKxTw0CwPuuZvVfnMoIRfpODRR
fVEk2VDor+lr8kqJkBaHgN5o/AvOXC7QCYadwbEkpr0ecxIZ1VcASYytIIM3YNL7
ZcHWwU5PCNLOdMXPqOdthhDhsHkKJNEXXr0YsjX/bQqYOUqYKPDyqh/yrrRO9Ro6
7eTSbfIguycAAwcH/2waLIQR8qYKxPknNuSdsOOqF2jf2gglL/7uMsIzjfkFzgHo
+GNHw9tmlZqD3yzaZ/N7Yv08ujRHhmWPBYAWRICBM3qo0zMJ9kI5XWRobeRQpLtf
YxxIOenq8R9t6YU9ryHdqf+P+Fi38eN5ERTDhNLrJOnO5/TA+of97BWCmdtJMlWM
RaHtqXxwo02Yi65IqKx6L7oOvT7Gh4NV2eglz8ZafPEoP8+V8ER7rwBYPiLk4Mse
oVImjveq2dmLUip9OPwznoaeyC8zB0mJ13m/KNC+CffkBgoXpMPiKzbu5YTjVw++
MlEmfgL42yDK0hnokmW2i9y1RBh/T1VQQeQbUaeITAQYEQIADAUCRM3ITgUJCWYB
gAAKCRCaemCth7qvrcbtAJ9j3C6lKNRB3uKcrfze66jAVQh0qACaAysOK82TcQ/2
73ryR0xWMFnpGqg=
=bMTb
-----END PGP PUBLIC KEY BLOCK-----
```

C. Licence

(## ####### #####) ## ######## ##### ### ##### ## #### ######## ####### ###### ####### (###### ## #########). ### #### ## ######### ## ######### ###/## #####

###. ### ### ## ### #### ##### #### ## ########## ##### #### ####### ## ######### ### ## ##########.

####, ### ###### ### ##### ## ## ##### ## ### ##### ## #### #######. ## ### ###### #### ####### ### ## ########## ## ## # ########, ### ######## ###### ### ### ###### ######### #### ## ############# ## #### ########## ## #### ##### ### ###########.

1. ###########

    1. ############ ##### # #### ##### #### ### ####, ## #### ### #### ### ##### ###–#####–### #####, #### ## # ############, ##########, ########## ####, ########### ## ##### ## ##### ########### ## # ######## ## ######## ####, ## ######### ## ########### ### ######## ################ ########### ## ### ##### #### ## ##### ### #### ### ## ######, ###########, ## ####### ######### ## ### #### ############ ####### #### ### ########, ###### #### # #### #### ########### # ########## #### ### ## ########## ## ########## ### ### ####### ## #### #######. ### ### ######### ## #####, ##### ### #### ## # ####### ####, ########### ## #########, ### ############### ## ### #### ## #####–######## #### # ###### ##### (##########) #### ## ########## ## ########## ### ### ####### ## #### #######.

    2. ############ ##### # ########### ## ######## ## ######## #####, #### ## #########–#### ### ###########, ## #############, ########## ## ##########, ## ##### ##### ## ####### ###### ##### #### ##### ###### ## ####### 1(#) #####, #####, ## ###### ## ### ######### ### ############ ## ##### ########, ########## ############ #########, ## ##### ### #### ## ######## ## ### ######## ## ########## #### ##### #### ### ## #### ##### ##############, #### ############ ######## ### ########### ##### ## ##########, ##### ######## ### ######### #### # ########## #####. # #### #### ###–######## # ########## #### ### ## ########## ## ########## (## ####### #####) ### ### ######## ## #### #######.

    3. ######### ####### ########## ######## ##### # ####### #### ## ###### ## ####://################.###/################## #### ### #### ######## ## ######## ####### ## ##### ########### ########## ## #### #######, #########, ## # #######, ####### #### #######: (#) ######## ##### #### #### ### #### #######, ####### ### ###### ## ### ####### ######## ## #### #######; ###, (##) ########## ####### ### ########### ## ########### ## ##### #### ######### ##### #### ####### ##### #### ####### ## # ###–##### ####### ############ ####### #### ### #### ####### ######## ## #### #######.

    4. ############ ##### ## #### ######### ## ### ###### ### ######## ### ###### ## ### #### ## ##########, ## ###########, ####### #### ## ##### ######## ## #########.

    5. ######## ######### ##### ### ######### ####–##### ####### ########## ## ######## ## ######## ### ######### ## ### ##### ## #### #######: ###########, ##########.

    6. ########## ##### ### ##########, ###########, ###### ## ######## #### #####(#) ### #### ##### ### ##### ## #### #######.

    7. ######### ####### #####, ## ### #### ## # ######## ## ######## ####, ### ##########, ###########, ###### ## ######## ### ####### ### #### ## ## ## ########## ## ###### ### ## ##########, ### #########; ### ## ######## (#) ## ### #### ## # ########### ### ######, #######, #########, #######, ### ##### ####### ### ###, ####, #######, #######, #### ##, ######### ## ######### ####### ######## ## ######## ##### ## ########### ## ########; (##) ## ### #### ## # ######### ### ######## ##### ### ###### ## ##### ###### ### ##### ##### ### ###### ## # ########### ## ##### ######; ###, (###) ## ### #### ## ##########, ### ############ #### ######### ### #########.

    8. ###### ##### ### ######## ###/## ######## #### ####### ##### ### ##### ## #### ##–##### ######### ####### ########## ### ########## ## ### #########, ########## ### ######## ######, ####### ### ## ### #### ## #### ## ### ########## ######## #######

####, #### ## # ####, ######## ### ##### #######; # #######, #######, ###### ## ##### #### ## ### #### ######; # ######## ## #########–####### ####; # ############# #### ## ############# ## #### ####; # ####### ########### #### ## ####### #####; # ####–########### #### ## ##### ### ########### ##### ######### ## # ####### ######### ## ##############; # #### ## #######, ########, ############, #########, ######### ## ###########; # ############ #### ## ##### ### ########### ##### ######### ## # ####### ######### ## ###########; # #### ## ####### ###; ## ############, ###, ####, ###### ## #####–########### #### ######## ## #########, ##########, ########### ## #######; # ###########; # #########; # #########; # ########### ## #### ## ### ##–#### ## ## ######### ## # ############# ####; ## # #### ######### ## # ####### ## ###### ######### ## ### ###### ## ## ### ######### ########## # ######## ## #####–### ####.

9. ##### ##### ## ########## ## ####### ########## ###### ##### #### ####### ### ### ### ########## ######## ### ##### ## #### ####### #### ####### ## ### ####, ## ### ### ######## ####### ########## #### ### ######## ## ######## ###### ##### #### ####### ####### # ######## #########.

10. ######### ######## ##### ## ####### ###### ########### ## ### #### ### ## #####–###### ## ### ###### ##### ###### ###########, ## ### ##### ## #######, ######### ## #### ## ######## ##### ## ###### ####### ############; ## #### ######### ## ### ###### ##### ## #### # ### #### ####### ## ### ###### ### ###### ##### ##### #### # ##### ### ## # ##### ########### ###### ## ####; ## ####### ### #### ## ### ###### ## ### ##### ## ####### ### ### ############# ## ### ###### ## ### ############ ## ### ####, ######### ## ###### ####### ###########; ## ######### ### ########### ### #### ## ### ##### ######### #####, ###### ## ######.

11. ########### ##### ## #### ###### ## ### #### ## ### ##### ######### ####### ####–###### ## ##### ## ###### ########## ### ### ##### ## ######## ### ########### ###–###### ## ### ####, ######### ####### ## # ######### ########### ## ######### ## ####### #### ## ##### ########## ######.

2. #### ####### ######. ####### ## #### ####### ## ######## ## ######, #####, ## ######## ### #### #### #### ######### ## ###### ####### #### ########### ## ########## #### ### ######## ### ## ########## #### ### ######### ########## ##### ######### ### ## ##### ########## ####.

3. ####### #####. ####### ## ### ##### ### ########## ## #### #######, ######## ###### ###### ### # #########, #######–####, ###–#########, ######### (### ### ######## ## ### ########## #########) ####### ## ######## ### ###### ## ### #### ## ###### #####:

   1. ## ######### ### ####, ## ########### ### #### #### ### ## #### ###########, ### ## ######### ### #### ## ########### ## ### ###########;

   2. ## ###### ### ######### ########### ######## #### ### #### ##########, ######### ### ########### ## ### ######, ##### ########## ##### ## ####### #####, ######### ## ######### ######## #### ####### #### #### ## ### ######## ####. ### #######, # ########### ##### ## ###### #### ######## #### ### ########## #### ####### ## ####–###,# ## # ############ ##### ######## #### ######## #### ### #### ########.#;

   3. ## ########## ### ######## ####### ### #### ######### ## ########### ## ######–#####; ###,

   4. ## ########## ### ######## ####### ###########.

   5. ### ### ######### ## #####:

      1. ###–######## ########## ####### #######. ## ##### ############# ## ##### ### ##### ## ####### ######### ####### ### ######### ## ########## ######### ######

   ###### ## ######, ### ######## ######## ### ######### ##### ## ####### #### ###–###### ### ### ######## ## ### ## ### ###### ####### ##### #### #######;

2. ######## ########## ####### #######. ## ##### ############# ## ##### ### ##### ## ####### ######### ####### ### ######### ## ########### ######### ###### ### ## ######, ### ######## ###### ### ######### ##### ## ####### #### ######### ### ### ######## ## ### ## ### ###### ####### ##### #### #######; ###,

3. ######### ####### #######. ### ######## ###### ### ##### ## ####### #########, ####### ############ ##, ## ### ##### #### ### ######## ## # ###### ## # ########## ####### #### ########### ######### ######### #######, ### #### #######, #### ### ######## ## ### ## ### ###### ####### ##### #### #######.

####–##### #######. ### ##### ###### ####### ### ##### ## #### #### ############# ## ### ########### ######### ## ######## ### ###### ## ##### ##### ### #######. ####### ## ####### 8(#), ### ###### ### ######### ####### ## ######## ### ###### ########.

4. ############. ### ####### ####### ## ####### 3 ##### ## ######### #### ####### ## ### ####### ## ### ######### ############:

   1. ### ### ########## ## ######## ####### ### #### #### ##### ### ##### ## #### #######. ### #### ####### # #### ##, ## ### ####### ######## ########## (###) ###, #### ####### #### ##### #### ## ### #### ### ########## ## ######## #######. ### ### ### ##### ## ###### ### ##### ## ### #### #### ######## ### ##### ## #### ####### ## ### ####### ## ### ######### ## ### #### ## ######## ### ###### ####### ## #### ######### ##### ### ##### ## ### #######. ### ### ### ########## ### ####. ### #### #### ###### ### ####### #### ##### ## #### ####### ### ## ### ########## ## ########## #### ##### #### ## ### #### ### ########## ## ######## #######. #### ### ########## ## ######## ####### ### ####, ### ### ### ###### ### ######### ############# ######## ## ### #### #### ######## ### ####### ## # ######### ## ### #### #### ### ## ######## ### ###### ####### ## #### ######### ##### ### ##### ## ### #######. #### ####### 4(#) ####### ## ### #### ## ############ ## # ##########, ### #### #### ### ####### ### ######### ##### #### ### #### ###### ## ## #### ####### ## ### ##### ## #### #######. ## ### ###### # ##########, #### ###### #### ### ######## ### ####, ## ### ###### ###########, ###### #### ### ########## ### ###### ## ######## ## ####### 4(#), ## #########. ## ### ###### ## ##########, #### ###### #### ### ######## ### ####, ## ### ###### ###########, ###### #### ### ########## ### ###### ## ######## ## ####### 4(#), ## #########.

   2. ### ### ########## ## ######## ####### ## ########## #### ##### ### ##### ##: (#) #### #######; (##) # ##### ####### ## #### ####### #### ### #### ####### ######## ## #### #######; (###) # ######## ####### ############ ####### (###### #### ## # ##### ####### #######) #### ######## ### #### ####### ######## ## #### ####### (#.#., ##–#########–########## 3.0 ##)); (##) # ######## ####### ########## #######. ## ### ##–##### ### ########## ##### ### ## ### ######## ######### ## (##), ### #### ###### #### ### ##### ## #### #######. ## ### ####### ### ########## ##### ### ##### ## ### ## ### ######## ######### ## (#), (##) ## (###) (### ########### ########), ### #### ###### #### ### ##### ## ### ########## ####### ######### ### ### ######### ##########: (#) ### #### ####### # #### ##, ## ### ### ###, ### ########## ####### #### ##### #### ## #### ########## ### ########## ## ######## #######; (##) ### ### ### ##### ## ###### ### ##### ## ### ########## #### ######## ### ##### ## ### ########## ####### ## ### ####### ## ### ######### ## ### ########## ## ######## ### ###### ####### ## #### ######### ##### ### ##### ## ### ########## #######; (###) ### #### #### ###### ### ####### #### ##### ## ### ########## ####### ### ## ### ########## ## ########## #### ##### #### ## ### #### ## ######## ## ### ########## ### ########## ## ######## #######; (##) #### ### ########## ## ######## ####### ### ##########, ### ### ### ###### ### ######### ############# ######## ## ### ########## #### ######## ### ####### ## # ######### ## ### ########## #### ### ## ######## ### ###### #######

#######. #### ####### 4(#) ##–##### ## ### ########## ## ############ ## # ##########, ### #### #### ### ####### ### ########## ##### #### ### ########## ###### ## ## #### ####### ## ### ##### ## ### ########## #######.

3. ## ### ##########, ## ######## ####### ### #### ## ### ########### ## ###########, ### ####, ###### # ####### ### #### #### ######## ## ####### 4(#), #### ###### ### ######### ####### ### ### #### ### #######, ########## ## ### ###### ## ##### ### ### #########: (#) ### #### ## ### ######## ###### (## #########, ## ##########) ## ########, ###/## ## ### ######## ###### ###/## ######## ######### ####### ##### ## ####### (#.#., # ####### #########, ########## ######, #######) ### ########### (###–######### ########) ## ########## ######### ######, ##### ## ####### ## ## ##### ########## #####, ### #### ## #### ##### ## #######; (##) ### ##### ## ### #### ## ########; (###) ## ### ###### ########## ###########, ### ###, ## ###, #### ######## ######### ## ## ########## #### ### ####, ###### #### ### #### ### ##### ## ### ####–##### ###### ## ######### ########### ### ### ####; ### (##) , ########## #### ####–#### 3(#), ## ### #### ## ## ###########, # ###### ########### ### ### ## ### #### ## ### ########## (#.#., ####### ########### ## ### #### ## ######## ######,# ## ########### ##### ## ######## #### ## ######## #######). ### ###### ######## ## #### ####### 4(#) ### ## ########### ## ### ########## ######; ########, #######, #### ## ### #### ## # ########## ## ###########, ## # ####### #### ###### #### ######, ## # ###### ### ### ############ ####### ## #### ########## ## ########## #######, #### ## #### ## ##### ####### ### ## # ###### ## ##### ## ######### ## ### ####### ### ### ##### #########–### #######. ### ### ######### ## #####, ### ### #### ### ### ####### ######## ## #### ####### ### ### ####### ## ########### ## ### ###### ### ### ##### ###, ## ########## #### ###### ##### #### #######, ### ### ### ########## ## ########## ###### ## ##### ### ########## ####, ########### ## ########### ## ### ######## ######, ######## ###/## ########### #######, ## ###########, ## ### ## #### ### ## ### ####, ####### ### ########, ####### ##### ####### ########## ## ### ######## ######, ######## ###/## ########### #######.

4. ###### ## ######### ###### ## ####### ## ### ######## ## ## ### ## ######### ###–###### ## ########## ###, ## ### #########, ########## ## ######## ####### ### #### ###### ## ###### ## ## #### ## ### ########### ## ###########, ### ##### ### #######, ########, ###### ## #### ##### ########## ###### ## ######## ## ### #### ##### ##### ## ########### ## ### ######## ######## ##### ## ##########. ######## ###### #### ## ##### ############# (#.#. #####), ## ##### ### ######## ## ### ##### ####### ## ####### 3(#) ## #### ####### (### ##### ## #### ###########) ##### ## ###### ## ## # ##########, ##########, ############ ## ##### ########## ###### ########### ## ### ######## ######## ##### ### ##########, ### ######## #### ##### ## ### ######, ## ###########, #### #######, ## ### ####### ###### ######### ## ### ########## ##–###### ###, ## ###### ### ## ########## ######## #### ##### ##### ####### 3(#) ## #### ####### (##### ## #### ###########) ### ### #########.

5. ###############, ########## ### ##########

#######, ######## ###### ### #### ##–## ### ##### ## ############### ## ########## ## ### #### ########## ### ####, ##–#####, #######, ######### ## #########, #########, ####### ##########, ########## ## #####, ###############, ####### ### # ########## #######, ###############, ## ### ##–##### ## ###### ## ##### #######, ########, ## ### ######## ## ####### ## ######, ####### ## ### ############. #### ############# ## ### ##### ### ######### ## ####### ###–#######, ## #### ######### ### ### ##### ## ###.

6. ########## ## #########. ###### ## ### ###### ######## ## ########## ###, ## ## ##### #### ######## ## ###### ## ### ## ### ##### ###### ### ### #######, ##########, #####–########, ######## ## ######### ####### ####### ### ## #### ####### ## ### ### ## ### ####, #### ## ######## ### #### ####### ## ### ########### ## #### #######.

7. ###########

   1. #### ####### ### ### ###### ####### ######### #### ######### ############ #### ### ###### ## ### ## ### ##### ## #### #######. ########### ## ######## ### #### ######## ########### ## ########### #### ### ##### #### #######, #######, #### ### #### ##### ######## ########## ######## #### ########### ## ######## ###### ## #### ######–#### #### ##### ########. ######## 1, 2, 5, 6, 7, ### 8 #### ####### ### ########### ## #### #######.

   2. ####### ## ### ##### ##### ### ##########, ### ####### ####### #### ## ######### (### ### ######## ## ### ########## ######### ## ### ####). ############### ### #####, ######## ######## ### ##### ## ####### ### #### ##### ######### ####### ##### ## ## ##### ############ ### #### ## ### ####; ########, ####### #### ### #### ######## #### ### ##### ## ######## #### ####### (## ### ##### ####### #### ### ####, ## ## ######## ## ##, ####### ##### ### ##### ## #### #######), ### #### ####### #### ######## ## #### ##### ### ###### ###### ########## ## ###### #####.

8. #############

   1. #### #### ### ########### ## ######## ####### ### #### ## # ##########, ### ######## ###### ## ### ######### # ####### ## ### #### ## ### #### ##### ### ########## ## ### ####### ####### ## ### ##### #### #######.

   2. #### #### ### ########### ## ######## ####### ## ##########, ######## ###### ## ### ######### # ####### ## ### ######## #### ## ### #### ##### ### ########## ## ### ####### ####### ## ### ##### #### #######.

   3. ## ### ######### ## #### ####### ## ####### ## ############# ##### ########## ###, ## ##### ### ###### ### ######## ## ############## ## ### ######### ## ### ##### ## #### #######, ### ####### ####### ###### ## ### ####### ## #### #########, #### ######### ##### ## ######## ## ### ####### ###### ######### ## #### #### ######### ##### ### ###########.

   4. ## #### ## ######### ## #### ####### ##### ## ###### ###### ### ## ###### ######### ## ###### #### ###### ## ####### ##### ## ## ####### ### ###### ## ### ##### ## ## ####### #### #### ###### ## #######.

   5. #### ####### ########### ### ###### ######### ####### ### ######## #### ####### ## ### #### ######## ####. ##### ### ## ###############, ########## ## ################ #### ####### ## ### #### ### ######### ####. ######## ##### ### ## ##### ## ### ##–######## ########## #### ### ###### ## ### ############# #### ###. #### ####### ### ### ## ######## ####### ### ###### ####### ######### ## ### ######## ### ###.

   6. ### ###### ####### #####, ### ### ####### ###### ##########, ## #### ####### #### ####### ######### ### ########### ## ### ##### ########## ### ### ########## ## ######## ### ######## ##### (## ####### ## ######### 28, 1979), ### #### ########## ## 1961, ### #### ######### ###### ## 1996, ### #### ############ ### ########## ###### ## 1996 ### ### ######### ######### ########## (## ####### ## #### 24, 1971). ##### ###### ### ####### ###### #### ###### ## ### ######## ############ ## ##### ### ####### ##### ### ###### ## ## ######## ######### ## ### ############# ########## ## ### ############## ## ##### ###### ########## ## ### ########## ######## ###. ## ### ######## ##### ## ###### ####### ##### ########## ######### ### ######## ########## ###### ### ####### ##### #### #######, #### ########## ###### ### ###### ## ## ######## ## ### #######; #### ####### ## ### ######## ## ######## ### ####### ## ### ###### ##### ########## ###.