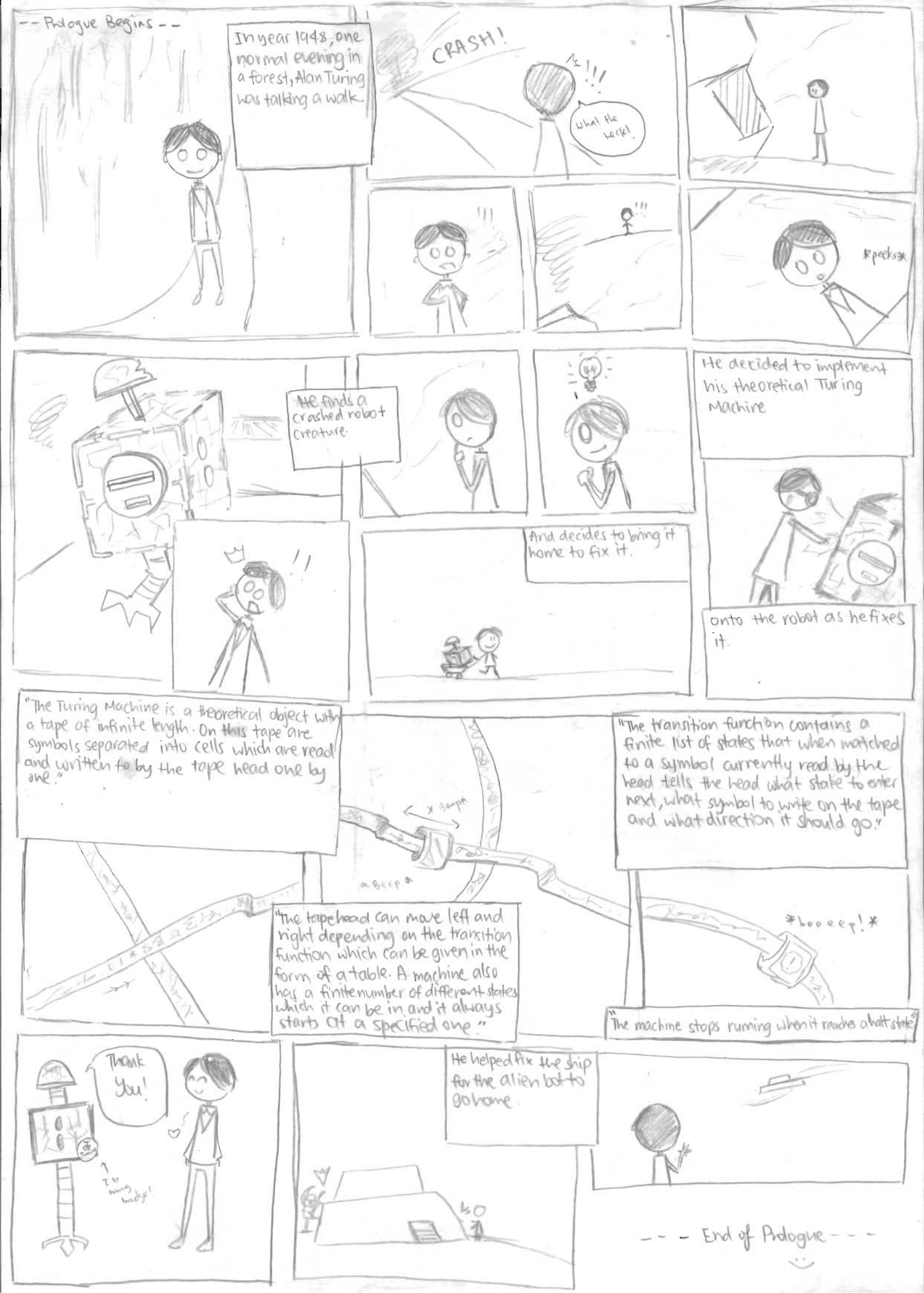
CS1231

Group 51

Names	Matriculation No
Eu Yong Xue	A0111995Y
Haritha Ramesh	A0115503W
Sharon Mariam Mathew	A0113000N
Jeremias Wong	A0066754W

The Halting Problem


-- Prologue Begins --
In year 1948, one normal evening in a forest, Alan Turing was talking a walk.
CRASH!
!!!
What the heck!
!!!
peeks
He finds a crashed robot creature.
He decided to implement his theoretical Turing Machine
And decides to bring it home to fix it.
onto the robot as he fixes it.
"The Turing Machine is a theoretical object with a tape of infinite length. On this tape are symbols separated into cells which are read and written to by the tape head one by one."
beep
"The transition function contains a finite list of states that when matched to a symbol currently read by the head tells the head what state to enter next, what symbol to write on the tape and what direction it should go."
"The tapehead can move left and right depending on the transition function which can be given in the form of a table. A machine also has a finite number of different states which it can be in and it always starts at a specified one."
booeep!
"The machine stops running when it reaches a halt state"
Thank You!
I'm coming home!
He helped fix the ship for the alien but to go home.
--- End of Prologue ---


Year 2051...
In a time of peace
Something despicable happened.
..... when technology was advanced.
ALERT!
spacestation, do you read me? There's something approaching
something big.....
what is happening?
WTH! It just appeared out of nowhere!
IMPENDING DOOM!
FOOSH!!!
AAAAARGHH!!!
AAARGHH!
PANIC
Chaos everywhere
BOOM!
BOOM!
BOOM!
KABOOM!!!
Massive destruction of human population


World as we knew it was gone
Mankind has conceded defeat.
Remaining survivors in hiding
Many travelled out of cities and into the wilderness to form refugee camps
It's only a matter of time before they decide to attack again.
Something has to be done about this...
There were rumours about a camp in Area 51
That there was a plan for rebellion. I was interested
Area 51
No trespass
I made my way to the camp
STOP
HALT!
Welcome to Area 51
... Another survivor!
They introduced me to the rest of them
Something weird happened though when I told them my name....
Hello everyone! I'm Nala Turing
. . .
His papers are widely acknowledged as the foundation of research in Artificial Intelligence
The central concept of modern computer was based on Turing's Paper
You can help us greatly in the rebellion!
Yes, I'll try
OMG IT'S THE DESCENDENT OF THE GREAT ALAN TURING
He is recognised as the great founder of Computer Science
proposed a "turing test" method to create an intelligence design standard for the tech industry

So it began, the discussions and stuff

Blah Blah

For a few days, we even sneaked in Alien lost parts to test.

Hmm... We don't know much about Turing's research. Most of it has been destroyed ...

Maybe we don't need his research. After all, we don't know much about the robots.

Even so, we don't really have an idea where to start, His work might be useful to us.

She really loves her van, doesn't she?

Guys! I have some of Alan Turing's Research in my family Diary

I forgot...

Why didn't you say so before???!

And so, a new more efficient research began...

She had an epiphany

I got it!

They came together to discuss their findings

Guys, can we maybe crash the robots' system?

How?

Make it run an infinite loop?

It might be able to detect the loop...

If the robots are built on a total functional programming language...

Ahh! Such a program will just not run!

What's that?

Total functional programming (TFP) differs from most "weak" forms by

⇒ restricting recursion to operating on reduced forms of its arguments.

⇒ forcing functions to be "total" i.e. have definitions for everything inside its domain

So this solves the problem of halting for functions right?

correct. However there are limitations of TFPs. Imagine the ticker tape of infinite length; such lengths does not exist in TFPs implementation of a machine!

So that means that TFPs are not Turing complete..

Guys...


The robots are based on the idea of Turing Machines*.
* refer to first page
That means programs with infinite loops do run inside them!!!!
I've got an idea!
while (true)
{ }
neverending loop!
I'm embarrassed to call you my brother
What?
I'm pretty sure such a loop can be detected by the robot
Even my grandma's compiler can detect this?
Okay what can we loop it with then?
Do you guys know what an undetectable problem is?
I was just suggesting
No...
Do you know what a decision problem is?
No!!!
Put simply, it is a problem with a yes or no answer
These guys are stupid
Ok...
An undetectable problem is a decision problem that cannot be solved correctly with a simple general algorithm
Ah... Are you thinking what I'm thinking?
What are you thinking?
evil mode
a:
goto a;
NO!
face palm
I'm thinking about the HALTING PROBLEM!
Woah.....


We know those bots are based off Turing machines...
Thanks to your ancestor...
Precisely! Now we have a way of stopping it!
The Halting Problem is an undecidable problem on Turing Machines
So that means we cannot make a single algorithm to tell us whether any program will halt?
Yes...
Now since we know the problem cannot be solved
We can trick the robot into trying to solve the Halting Problem....
Smart bro!
But what if it already knows about the Halting Problem...
We'll unnecessarily endanger the lives of our greatest minds!
let's see...
Click
1948-
Alan just came back from fixing the robot
THE HALTING PROBLEM
OH NO!
I forgot to tell him/her about the Halting Problem
!
HALTING
I do hope it won't bother him in future much...
Oh well...
Present Day
The Robot doesn't know about the Halting Problem!
That settles it! We can trick it and defeat it!
Let's get to work people!
The resistance starts getting ready at a location away from area 51.
Communication device
Communications was established!
Talk to me


Silly human. By contacting me you have exposed your position to me....
I know, just wanted to chat...
?!
Go on then, amuse me.
You certainly will be..
Tell me, do you think you can solve every problem I give you?
Of course! I'm the motherboard, the cleverest being around here, and the rest of the universe!
I have conquered 89% of the universe, how dare you challenge my intellect?
Brawn ≠ Brain, Motherboard
Oh, is that so?
The robots have found her!
Since you are going to kill me, why don't you answer a problem
I really want to know the answer,
Oh alright, why don't you ask away
Can you find me a simple algorithm that can tell if any program will halt or run forever?
Sure.. Here you go.
WTH!?
!!
if(program_contains("while(true)" or "for(;;)") return false else return true
This is utter nonsense
Can I kill you now?
You know as well as I do that, it is not enough!
So what?
If you don't solve it you can't claim to be the cleverest around here..
Alright.. I'll do it
Don't you try to escape or anything, my robots are watching...
Don't you worry. I'll be right here.
Processing...
...
Processing...
You have any "twos"?
Back in area 51
It's been hours since Nalalei left. Do you think she's fine?
I'm sure she can handle herself, mate.
yes, such great courage to face the motherboard on her own,
she'll be fine dude

1. We want to prove that a function $H(x,y)$ that checks if a function x given an input y will halt does not exist, in other words:

2. Let $H(x,y)$ be a function such that x and y are functions and

$$\forall x \forall y \left(\left(\text{time}(x(y)) < \infty \iff H(x,y) = 1 \right) \wedge \left(\text{time}(x(y)) = \infty \iff H(x,y) = 0 \right) \right)$$

3. Let us assume H exists

1. Let $R(x)$ be a function that x is a function and $\forall x (H(x,x) = 1 \Leftrightarrow \text{time}(R(x)) = \infty) \wedge (H(x,x) = 0 \Leftrightarrow \text{time}(R(x)) < \infty)$
2. We let $x = R$ such that $R(R)$

3. Let us assume $\text{time}(R(R)) = \infty$
   a. $H(R,R) = 1$ by 3.1
   b. $\text{time}(R(R)) < \infty$ by 2
   c. $\text{time}(R(R)) = \infty \wedge \text{time}(R(R)) < \infty$
   d. This is a contradiction.

d

4. $\text{time}(R(R)) \neq \infty$

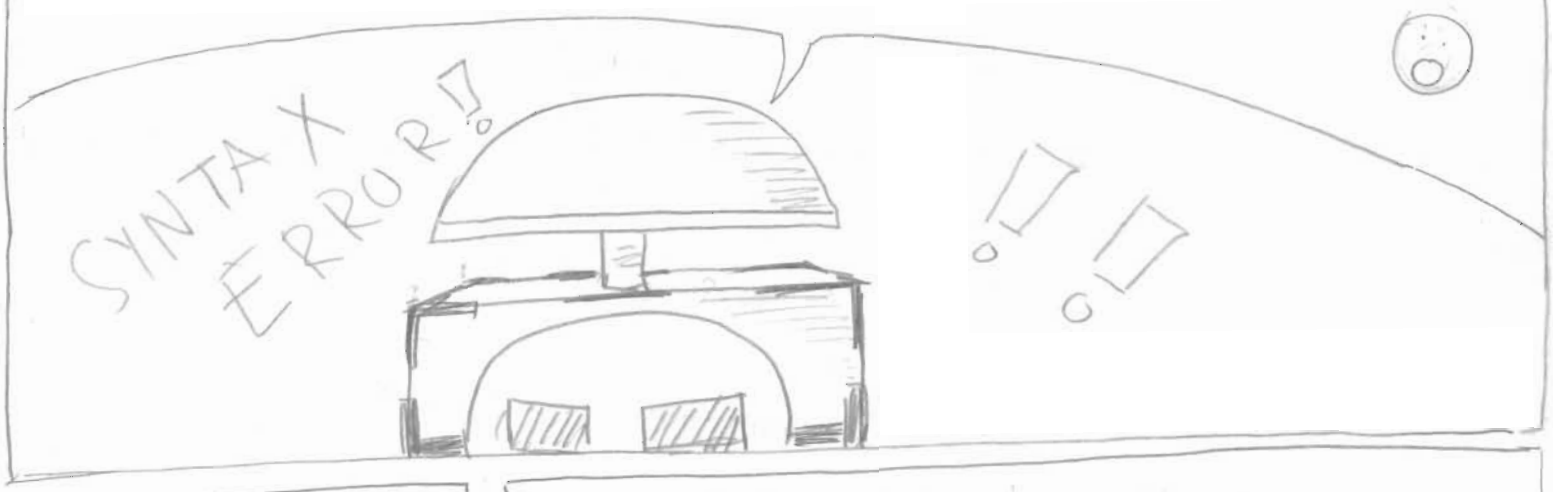
5. Let us assume $\text{time}(R(R)) < \infty$.
   a. $H(R,R) = 0$ by 3.1
   b. $\text{time}(R(R)) = \infty$ by 2
   c. $\text{time}(R(R)) = \infty \wedge \text{time}(R(R)) < \infty$
   d. This is a contradiction.

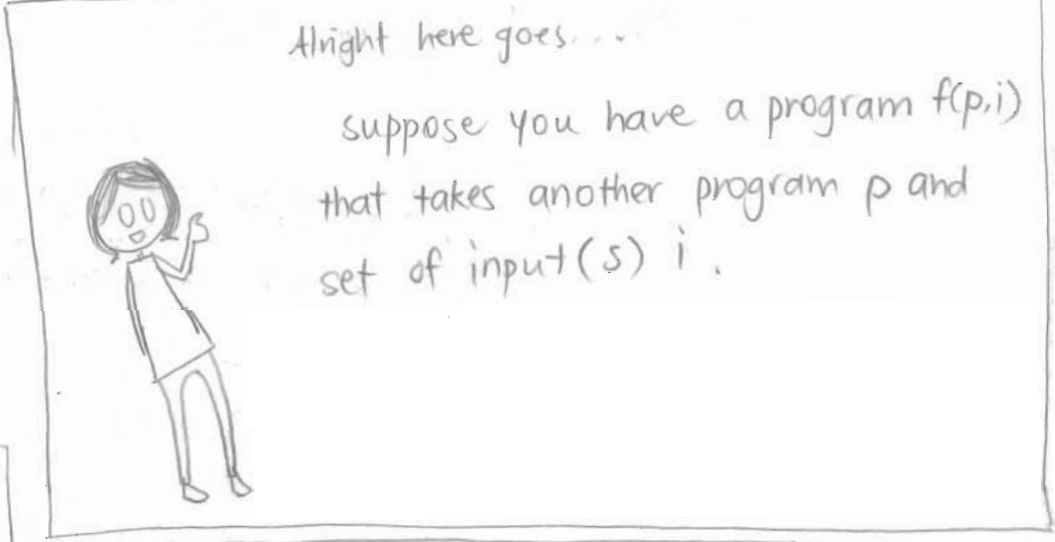
6. $\text{time}(R(R)) = \infty$
7. $\text{time}(R(R)) = \infty \wedge \text{time}(R(R)) \neq \infty$ by 3.4 and 3.6
8. This is a contradiction

4. H does not exist

Q.E.D

BUT I DO NOT UNDERSTAND THE SYNTAX OF YOUR LANGUAGE!!





such that

f(p,i) returns 1 if the program halts and 0 otherwise.

suppose we have the partial function* g(p) that returns 0 if f(p,p) == 0 and executes an infinite loop otherwise...

ie

```
procedure compute_g(i)
    if f(i,i) == 0 then
        return 0
    else
        while (i)
            pass
```

NOTICE THAT

g is partially computable

⇓

∃e such that e is a program that computes the partial function g (ie. e == compute_g)

Let's show that $f(p,i)$ is not an arbitrary total function*
by proving that $\exists k_1, \exists k_2, \exists k_3 : f(k_1) = k_2 \wedge f(k_1) = k_3 \wedge (k_2 \neq k_3)$

Assume $f(e,e) \geq 0$

then compute_g(e) = e(e) = 0

then $f(e,e) = 1$ $(\Rightarrow\Leftarrow)$

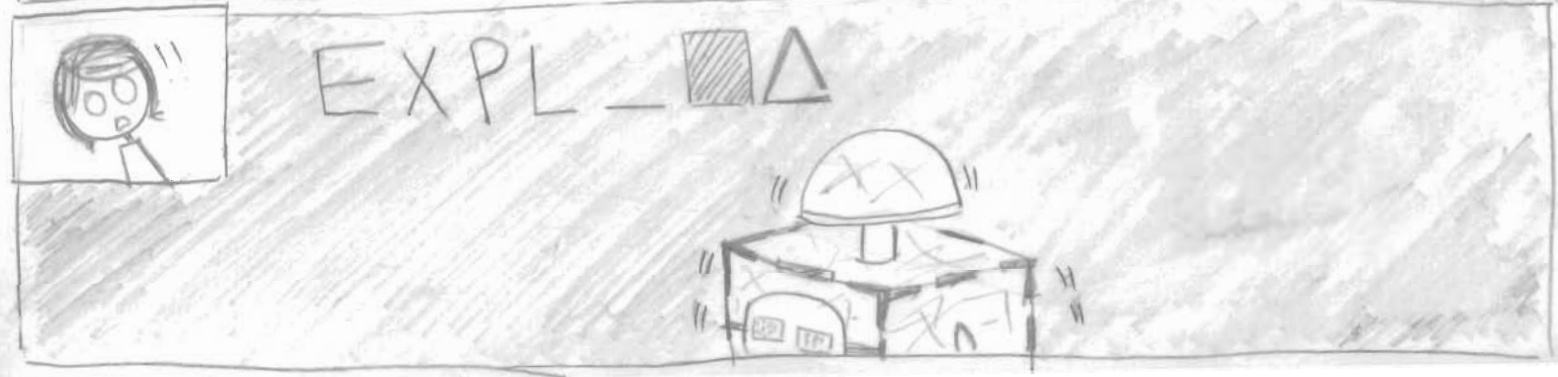
Assume $f(e,e) = 1$,

then compute_g(e) = e(e) does not halt

then $f(e,e) = 0$ $(\Rightarrow\Leftarrow)$

Thus $f(p,i)$ is not an arbitrary total function ∎

* partial computable function, p, is a function whose domain may not be defined for every possible input.
* total computable function, Q, is a function that is defined for all possible inputs




KABOOM
...
We did it...
We did it!!!
And so they rebuilt the world with the help of the leftover minion bots.
Simple programming was all it took.
Slowly, the world they loved and treasured was coming back again.
We'll help!!
Other survivors joined and helped.
They crowned Nala saviour of the world!
NALA TURING
So the Halting Problem huh
turn
Wow. Oh yeah.
Yeah that's the reason why we have human QA testers (quality assurance) to check the programs.
You really are the descendant of the great Alan Turing
He he

FIN

References:

Chu-Carroll, M. C. (2007, February 3). *Basics: The Turing Machine (with an interpreter!)*. Retrieved from Science Blogs: http://scienceblogs.com/goodmath/2007/02/03/basics-the-turing-machine-with-1/

De Heredia, J. B. (2012, May 4). *The undecidability of the halting problem is not very important.* Retrieved from jonbho: http://jonbho.net/2012/05/04/the-undecidability-of-the-halting-problem-is-not-very-important/

Hehner, E. C. (2012, October 15). *Problems with the Halting Problem.* Retrieved from University of Toronto Department of Computer Science: http://www.cs.utoronto.ca/~hehner/PHP.pdf

Hodges, A. (1995). *Alan Turing: A Short Biography*. Retrieved from Alan Turing: the enigma: http://www.turing.org.uk/bio/part1.html

Hrbacek, K., & Jech, T. (1999, June 22). *Introduction to Set Theory, Revised and Expanded*. Crc Press.

Pullum, G. K. (2008). *Scooping the Loop Snooper.* Retrieved from Linguistics and English Language, The University of Edinburgh: http://www.lel.ed.ac.uk/~gpullum/loopsnoop.html

Total Functional Programming. (n.d.). In Wikipedia. Retrieved October 26, 2013 from http://en.wikipedia.org/wiki/Total_functional_programming