Zhang Mohan 1002748716

Discussed with Ai Yifei

No outside material consulted

Proof for Question No.1:

Suppose there exist a function C(Q), that, given input Q, outputs:

T, if Q is a syntactically correct Python program, there is some input on which Q halts, and there is some input on which Q does not halt.

F, otherwise.

```
def HHH(M):

    if C(HHH):

        while True:

            pass

    elif M == “1”:

        while True:

            pass

    elif M == “2”:

        return 240
```

If C(HHH) = T: For HHH(M), whatever M is, HHH(M) won't halt. So, C(HHH) should be F.

If C(HHH) = F: For HHH(M), If M = “2”, HHH(M) halts. If M == “1”, HHH(M) won’t halt. There is some input on which Q halts, and there is some input on which Q does not halt. So, C(HHH) should be T. (reduction part)

Then this HHH(M) is a contradiction. Thus this kind of C(Q) does not exist.

Proof for Question No.2:

2.(a)

a_n	b_n	c_n	s_n	c_{n+1}	$a_n + b_n + c_n$	$2c_{n+1} + s_n$	$a_n + b_n + c_n = 2c_{n+1} + s_n$
0	0	0	0	0	0	0	T
1	0	0	1	0	1	1	T
0	1	0	1	0	1	1	T
0	0	1	1	0	1	1	T
1	1	0	0	1	2	2	T
1	0	1	0	1	2	2	T
0	1	1	0	1	2	2	T
1	1	1	1	1	3	3	T

2.(b)

Prove by induction:

1.Suppose $a_n + b_n + c_n = 2c_{n+1} + s_n$ for all n ϵ N

Base case:

2.for n = 0, $\mathrm{num}(\alpha_n) = \mathrm{a}_0$, $\mathrm{num}(\beta_n) = \mathrm{b}_0$, $\mathrm{num}(\sigma_n) = s_0$

3. $\mathrm{num}(\alpha_n) + \mathrm{num}(\beta_n) + c_0 = \mathrm{a}_0 + \mathrm{b}_0 + c_0 = 2c_1 + s_0 = 2^1 c_1 + \mathrm{num}(\sigma_n)$
(directly prove from 1 and 2)

Induction step:

3. let n ϵ N be arbitrary,

5. Assume $\mathrm{num}(\alpha_n) + \mathrm{num}(\beta_n) + c_0 = 2^{n+1}c_{n+1} + \mathrm{num}(\sigma_n)$

6. $\mathrm{num}(\alpha_{n+1}) = a_{n+1}\, 2^{n+1} + \mathrm{num}(\alpha_n), \mathrm{num}(\beta_{n+1}) = b_{n+1}\, 2^{n+1} + \mathrm{num}(\beta_n),$
$\mathrm{num}(\sigma_{n+1}) = s_{n+1}\, 2^{n+1} + \mathrm{num}(\sigma_n)$ (by definition of binary representation of integers)

7. $\mathrm{num}(\alpha_{n+1})+ \mathrm{num}(\beta_{n+1}) + c_0 = a_{n+1}\, 2^{n+1} + \mathrm{num}(\alpha_n) + b_{n+1}\, 2^{n+1} +$
$\mathrm{num}(\beta_n) + c_0$

(directly prove from 6)

$= a_{n+1}\, 2^{n+1} + b_{n+1}\, 2^{n+1} + 2^{n+1}c_{n+1} + \mathrm{num}(\sigma_n)$

(directly prove from 5)

$=2^{n+1}(2c_{n+2} + s_{n+1} - c_{n+1}) + 2^{n+1}c_{n+1} + \mathrm{num}(\sigma_n)$

(directly prove from 1)

$= 2^{n+2}c_{n+2} + 2^{n+1}s_{n+1} + \mathrm{num}(\sigma_n)$

$= 2^{n+2}c_{n+2} + \mathrm{num}(\sigma_{n+1})$ (by definition of binary representation of integers)

8. $\mathrm{num}(\alpha_n) + \mathrm{num}(\beta_n) + c_0 = 2^{n+1}c_{n+1} + \mathrm{num}(\sigma_n)$ *IMPLIES*

$\mathrm{num}(\alpha_{n+1}) + \mathrm{num}(\beta_{n+1}) + c_0 = 2^{n+2}c_{n+2} + \mathrm{num}(\sigma_{n+1})$ (directly prove from 7)

9. $\forall n \in N.\ \mathrm{num}(\alpha_n) + \mathrm{num}(\beta_n) + c_0 = 2^{n+1}c_{n+1} + \mathrm{num}(\sigma_n)$ *IMPLIES*

$\mathrm{num}(\alpha_{n+1}) + \mathrm{num}(\beta_{n+1}) + c_0 = 2^{n+2}c_{n+2} + \mathrm{num}(\sigma_{n+1})$ (by generalization)

10. $\forall n \in N.\ \mathrm{num}(\alpha_n) + \mathrm{num}(\beta_n) + c_0 = 2^{n+1}c_{n+1} + \mathrm{num}(\sigma_n)$ (by induction)