۳) a) expression = a[redacted]

این الگوریتم به این صورت کار می‌کند که k را تبدیل به مبنای دو کرده

(دودویی می‌کند) [redacted]،

و در هر مرحله عبارت را به توان ۲ رسانده و در صورتی

که در طول پیمایش عدد دودویی k با یک بیت ۱ مواجه

شود آن را در a نیز ضرب می‌کند.

پیمایش ارقام عدد دودویی k

- مواجه با ۰ : $y \leftarrow y \times y$ [redacted] $\times a$
- مواجه با ۱ : $y \leftarrow y \times y$

۳) b) در هر حلقه‌ی while مقدار i که ابتدا با k شروع می‌شود نصف شده و ۱ عمل (در هر تکرار) ($O(1)$) نیز اجرا می‌شود. (زیرا همه‌ی ضرب‌ها و تقسیم‌ها را می‌توان $O(1)$ حساب کرد.) بنابراین این حلقه $\log(k)$ بار اجرا می‌شود.

$$\sum_{i=1}^{\log(k)} O(1) = O(\log k)$$

c) در هر تکرار از حلقه (تکرار p ام)، a^{α} که a عددی معادل مقدار p رقم سمت راست نمایش دودویی k است حساب شده است.

برای اثبات ثابت بودن این گزاره در هر تکرار کافی است برای شروع آن $p=1$ حساب کنیم که درستی آن بدیهی است (اگر عدد فرد، رقم سمت راست برابر ۱ و برای عدد زوج این رقم ۰ است.

حال فرض می‌کنیم که در حلقه‌ی p ام این رابطه درست است.

در تکرار بعدی حلقه i که برابر مقدار عدد از رقم i ام نمایش دودویی k به بعد است چک می‌شود زوج است یا نه. یعنی رقم i ام باینری k ۱ است یا ۰. اگر برابر ۰ باشد a ثابت می‌ماند (زیرا رقم ۰ در سمت راست تأثیری ندارد) و اگر برابر ۱ باشد مقدار a، x^{p+1} برابر می‌شود. یعنی مقدار معادل p رقم سمت راست باینری k، 2^p اضافه می‌شود. بنابراین

در ذیل حاله چال حکم مورد نظر ارفاقی صادر.