نکاتی برای برنامه‌نویسان [1]

ترجمهٔ وحید مواجی

آذر ۱۳۸۴

[1] From The Pragmatic Programmer by Andrew Hunt & David Thomas

نکته ۱، به کار خود توجه نشان دهید

اگر برایتان مهم نیست که کارتان را درست انجام دهید، چرا عمر خود را با نوشتن نرم‌افزار تلف می‌کنید؟

نکته ۲، دربارهٔ کارتان فکر کنید!

کار طوطی‌وار را رها کرده و آگاهانه عمل کنید. به طور مداوم کار خود را نقد و ارزیابی کنید.

نکته ۳، راه چاره‌ای بیندیشید، عذر بدتر از گناه نیاورید

به جای معذرت‌خواهی، راه چاره‌ای بیندیشید. نگویید که این کار امکان‌پذیر نیست؛ بگویید که چه چیزی امکان‌پذیر است.

نکته ۴، دست به عصا راه نروید

طراحی‌های بد، تصمیمات نادرست و برنامه‌های ضعیف را به محض مشاهده درست کنید.

نکته ۵، تسریع‌دهندهٔ تغییرات باشید

نمی‌توانید بقیه را وادار به تغییرات کنید. بلکه به آنها نشان دهید که آینده چگونه خواهد بود و به آنها در ساختن آن آینده کمک کنید.

نکته ۶، اصل قضیه را فراموش نکنید

آن قدر در جزئیات غرق نشوید که فراموش کنید در اصل می‌خواستید چه کار کنید.

نکته ۷، کیفیت را سرلوحهٔ کار خود قرار دهید

کاربرانِ خود را در تعیینِ نیازمندی‌های کیفیتیِ واقعیِ سیستم سهیم سازید.

نکته ۸، همواره برای افزایش دانش خود هزینه کنید

به آموختن عادت کنید.

نکته ۹، آن چه می‌خوانید و می‌شنوید را با نگاه نقادانه تحلیل کنید

فریفته و سرگردانِ فروشنده‌ها، رسانه‌ها و یا اسیر تعصبات نشوید. تمامیِ اطلاعات را از نقطه‌نظرِ خود و پروژهٔ خود تحلیل کنید.

نکته ۱۰، هم چیزی که می‌گویید مهم است، هم این که آن را چگونه می‌گویید

داشتن ایده‌های بزرگ فایده‌ای ندارد اگر ندانید چگونه آنها را به دیگران انتقال دهید.

نکته ۱۱، در ورطهٔ تکرار نیفتید

هر دانشی باید یک نمودِ عینیِ واحد، واضح و معتبر در سیستم داشته باشد.

نکته ۱۲، استفادهٔ مجدد را آسان سازید

فقط اگر استفادهٔ مجدد آسان باشد، مردم این کار را می‌کنند. محیطی برای استفادهٔ مجدد فراهم سازید.

نکته ۱۳، چیزهای نامرتبط نباید بر هم تأثیری بگذارند

اجزایی را طراحی کنید که متکی به خود و مستقل باشند و هدفِ واحد و تعریف‌شده‌ای داشته باشند.

نکته ۱۴، هیچ تصمیمی نهایی نیست

هیچ تصمیمی برای همیشه پابرجا نیست. بلکه فکر کنید همهٔ تصمیم‌ها را روی شنِ دریا نوشته‌اید و برنامه‌ای برای تغییراتِ مناسب داشته باشید.

نکته ۱۵، برای پیدا کردنِ هدف از منوّر استفاده کنید

منوّرها به شما اجازه می‌دهند که با سعی‌های مکرر در تاریکی ببینید چقدر از هدف خود فاصله دارید.

نکته ۱۶، از مدلِ اولیه برای یادگیری استفاده کنید

مدل‌سازیِ اولیه، یک روش یادگیری است. ارزش آن فقط در برنامه‌ای که می‌نویسید نیست بلکه در درس‌هایی است که از آن می‌آموزید.

نکته ۱۷، برنامهٔ خود را به مسألهٔ واقعی نزدیک سازید

به زبان کاربران خود طراحی و برنامه‌نویسی کنید.

نکته ۱۸، برای غافلگیر نشدن، کارها را ارزیابی کنید

قبل از این که شروع کنید، همه چیز را ارزیابی کنید. در این صورت مشکلاتِ احتمالی را همان اول مدّ نظر خواهید داشت.

نکته ۱۹، زمان‌بندیِ برنامه را بررسی کنید

از تجربه‌ای که هنگام پیاده‌سازی برنامه به دست می‌آورید برای ویرایشِ زمان‌بندیِ پروژه استفاده نمایید.

نکته ۲۰، دانشِ خود را به متنِ ساده نگهداری کنید

متن ساده هیچ‌گاه منسوخ نخواهد شد. بلکه کمک می‌کند تا در کارِ خود مسلط شوید و همچنین باگ‌زدایی و آزمایش برنامه را ساده‌تر خواهد ساخت.

نکته ۲۱، از قدرتِ کنسول‌های فرمان بهره گیرید

وقتی که واسط‌های گرافیکی جواب‌گو نمی‌باشند، از کنسول‌های متنی استفاده کنید.

نکته ۲۲، به خوبی از یک ویرایشگر متن استفاده کنید

ویرایشگر متنِ موردِ نظر باید مثل موم در دستان شما باشد؛ مطمئن شوید که ویرایشگر شما قابل‌تنظیم و قابل‌گسترش است و قابلیتِ برنامه‌نویسی دارد.

نکته ۲۳، همیشه از سیستم‌های کنترل کُد منبع استفاده کنید

سیستم کنترل کُد منبع به مثابه یک ماشین زمان می‌ماند که می‌توانید تغییرات کارِ خود را در زمان نگهدارید.

نکته ۲۴، مشکل را حل کنید، سرزنش نکنید

واقعاً اهمیتی ندارد که ایراد برنامه تقصیر شماست یا کَسِ دیگری. این مسأله کماکان مشکل شماست و باید توسط شما رفع گردد.

نکته ۲۵، موقع باگ‌زداییِ برنامه، پریشان نشوید

یک نفس عمیق بکشید و دربارهٔ علت مشکل فکر کنید!

نکته ۲۶، فرافکنی نکنید

بسیار به ندرت اتفاق می‌افتد که مشکلی در سیستم‌عامل یا کامپایلر یا حتی محصولِ جانبی که خریده‌اید وجود داشته باشد. اغلب اوقات مشکل از برنامهٔ خودتان است.

نکته ۲۷، نظریه‌پردازی نکنید، ادعای خود را اثبات کنید

نظریه‌های خود را در محیط واقعی با داده‌ها و محدودیت‌های واقعی اثبات کنید.

نکته ۲۸، یک زبان کار با متن بیاموزید

قسمت عمده‌ای از هر روزِ خود را به کار با متن می‌گذرانید. چرا نمی‌گذارید که قسمتی از این کار را کامپیوتر برای شما انجام دهد؟

نکته ۲۹، برنامه‌ای بنویسید که خودش برنامه بنویسد

تولیدکننده‌های خودکارِ برنامه، بازدهی شما را افزایش داده و مانعِ دوباره‌کاری می‌شوند.

نکته ۳۰، نمی‌توانید نرم‌افزاری عالی بنویسید

نرم‌افزار نمی‌تواند عالی و بی‌عیب باشد. کاری کنید که برنامهٔ شما و کاربرانتان از خطاهای اجتناب‌ناپذیر محافظت شوند.

نکته ۳۱، طبق قرارداد طراحی کنید

برای مستندسازی از قرارداد استفاده کنید و مطمئن شوید که برنامهٔ شما چیزی بیشتر یا کمتر از چیزی که ادعا می‌کند، انجام ندهد.

نکته ۳۲، هر چه زودتر برنامه را تا حد مرگ تحت فشار بگذارید

یک برنامهٔ مُرده معمولاً بسیار کمتر از یک برنامهٔ چلاق، آسیب می‌رساند.

نکته ۳۳، از روش‌های مطمئن‌سازی برای جلوگیری از روز مبادا استفاده نمایید

روش‌های مطمئن‌سازی، اعتبار فرضیات شما را می‌سنجند. از آنها برای محافظتِ برنامهٔ خود در برابر دنیای نامطئن استفاده کنید.

نکته ۳۴، برای مواردِ استثنایی از استثنائات استفاده نمایید

استثنائات می‌توانند از همهٔ مشکلاتِ مربوط به خوانایی و نگهداری که در برنامه‌های قدیمی وجود داشت رنج ببرند. استثنائات را برای مواردِ استثنایی بگذارید.

نکته ۳۵، آن چه را که شروع می‌کنید به پایان برسانید

تا آنجا که ممکن است، هر رویه یا شیئ‌ای که منبعی را در اختیار می‌گیرد باید بعد از اتمامِ کارش آن را آزاد سازد.

نکته ۳۶، اتصال بین ماژول‌ها را به حداقل برسانید

از نوشتن برنامه‌های «کم‌رو» که متکی بر دیگر برنامه‌ها هستند اجتناب کنید.

نکته ۳۷، پیکربندی کنید نه این که همه چیز را ثابت قرار دهید

دست کاربر را در انتخاب تنظیمات مختلف برای نرم‌افزار باز بگذارید نه این که همه چیز را به طور ثابت در نرم‌افزار تعبیه سازید.

نکته ۳۸، تجرید را در برنامه قرار دهید، جزئیات را در فایل‌های جانبی

برای حالت کلی برنامه بنویسید و جزئیاتِ حالت‌های خاص را در فایل‌های جانبی و بیرون از برنامه قرار دهید.

نکته ۳۹، جریان‌کار را برای بهبود هم‌زمانی مورد بررسی قرار دهید

از جریان‌کارِ کاربرانِ خود، هم‌زمانی‌های ممکن را استخراج کنید.

نکته ۴۰، با استفاده از سرویس‌ها طراحی کنید

با استفاده از سرویس‌ها، پیاده‌سازی‌ها را پُشتِ واسط‌هایی که به خوبی طراحی و تعریف شده‌اند، پنهان سازید.

نکته ۴۱، همیشه هم‌زمانی را هنگام طراحی درنظر داشته باشید

اگر فرض را بر هم‌زمانی بگذارید، واسط‌های بهتری طراحی خواهید کرد و فرضیات کمتری خواهید داشت.

نکته ۴۲، ویوها را از مدل جدا سازید

با طراحی بر مبنای مدل-ویو، با هزینه‌ای کم، انعطاف‌پذیریِ زیادی به نرم‌افزارِ خود می‌بخشید.

نکته ۴۳، برای هماهنگ کردن جریان‌کارها از تخته‌سیاه استفاده نمایید

از تخته‌سیاه برای هماهنگیِ فَکت‌ها و عواملِ مختلف استفاده نمایید و درعین‌حال استقلالِ موجود بین همکاران را حفظ کنید.

نکته ۴۴، الله‌بختکی برنامه ننویسید

فقط به چیزهای مطمئن اتکا کنید. از پیچیدگیِ تصادفی آگاه باشید و یک اتفاقِ خوشحال‌کننده را در یک برنامه‌ریزیِ هدفمند وارد نسازید.

نکته ۴۵، مرتبهٔ الگوریتمِ خود را تخمین بزنید

قبل از این که برنامه‌ای بنویسید، ارزیابی کنید که آن برنامه چقدر طول می‌کشد تا اجرا شود.

نکته ۴۶، تخمین‌های خود را بیازمایید

تحلیل ریاضیِ الگوریتم‌ها همه چیز نیست. حتماً برنامهٔ خود را در محیط واقعی اجرا کرده و زمانِ اجرای آن را در نظر بگیرید.

نکته ۴۷، برنامهٔ خود را همیشه تمیز نگه دارید

همان طور که ممکن است باغچهٔ خود را وِجین کرده و علف‌های هرز را قطع کنید، هر زمانی که برنامهٔ شما نیاز دارد، آن را دوباره بنویسید، دوباره انجام دهید و در معماریِ آن تجدیدنظر کنید. همیشه سعی کنید ریشهٔ مشکلات را قطع کنید.

نکته ۴۸، موقع طراحی، تست را فراموش نکنید

قبل از این که حتی یک خط برنامه بنویسید، به روشِ تست و آزمایشِ آن بیندیشید.

نکته ۴۹، یا خودتان نرم‌افزار را تست کنید یا این که کاربرانِ عصبانی این کار را خواهند کرد

برنامهٔ خود را بی‌رحمانه تست کنید تا به کاربران اجازه ندهید اشکالاتِ آن را برایتان پیدا کنند.

نکته ۵۰، از برنامه‌های ویزاردی که عملکردِ آنها را نمی‌فهمید استفاده نکنید

ویزاردها می‌توانند یک مثنویِ هفتادمن‌کاغذ از برنامه تولید کنند. قبل از این که برنامهٔ تولیدشده را وارد نرم‌افزارِ خود کنید مطمئن شوید که روشِ کارِ آن را فهمیده‌اید.

نکته ۵۱، نیازمندی‌ها را به طور عمقی مورد بررسی قرار دهید

به‌ندرت پیش می‌آید که نیازمندی‌های کاربر همان چیزی باشد که می‌گوید، نیازهای واقعی او زیر یک خروار فرضیات، سوتفاهمات و رِندی‌ها پوشیده شده است.

نکته ۵۲، با یک کاربر کار کنید تا مثل یک کاربر بیندیشید

این بهترین روشی است که درخواهیدیافت نرم‌افزارِ شما چگونه مورد استفاده قرار خواهد گرفت.

نکته ۵۳، تجریدها بیشتر از جزئیات عمر می‌کنند

روی تجرید هزینه کنید نه روی پیاده‌سازی. تجریدها زیرِ رگبارِ تغییراتِ پیاده‌سازی و تکنولوژی‌های جدیدی که می‌رسند به حیاتِ خود ادامه می‌دهند.

نکته ۵۴، برای پروژهٔ خود یک واژه‌نامه فراهم سازید

تمام اصطلاحات و واژگانِ خاصّی که در پروژهٔ شما مورد استفاده قرار می‌گیرند باید در یک مستند شرح داده شوند.

نکته ۵۵، خارج از گود ننشینید، وارد گود شوید

وقتی با یک مسألهٔ لاینحل مُواجِه می‌شوید، محدودیت‌های اصلی را مشخص سازید. از خود بپرسید: آیا باید این طور انجام شود؟ آیا اصلاً باید انجام شود؟

نکته ۵۶، وقتی شروع کنید که آماده هستید

همیشه از تجربه درس بگیرید. هیچ وقت شک‌های آزاردهنده را کم‌اهمیت تلقی نکنید.

نکته ۵۷، بعضی از چیزها را نباید توضیح داد، باید انجام داد

وقتی لازم است که شروع به برنامه‌نویسی کنید، در منجلابِ مستندسازی نیفتید.

نکته ۵۸، بردهٔ تقلید نباشید

تا زمانی که یک تکنولوژی را در محیط پروژهٔ خود و تحت قابلیت‌های نرم‌افزارِ خود مورد آزمایش قرار نداده‌اید، هیچ‌گاه کورکورانه از آن استفاده نکنید.

نکته ۵۹، ابزارهای گران‌قیمت منجر به طراحی‌های بهتری نمی‌شوند

فریفتهٔ شعارهای تبلیغاتی، قیمت‌های گزاف و قضاوت دیگران نشوید. ابزارهای طراحی خود را با معیارهای مناسب و مفید به پروژهٔ خود بسنجید.

نکته ۶۰، تیم‌های خود را بر اساس عملکرد، سازماندهی کنید

طراح‌ها را از برنامه‌نویسان و تست‌کننده‌ها را از مدل‌گرها جدا نسازید .تیم‌ها را براساس کاری که انجام می‌دهند و هدف نهایی که به آن خواهند رسید بچینید.

نکته ۶۱، از رویه‌های دستی استفاده نکنید

یک اسکریپت یا فایلِ دسته‌ای همان دستورات را به همان ترتیب و پشتِ سرِ هم انجام می‌دهد.

نکته ۶۲، زود تست کنید، همیشه تست کنید. به طور خودکار تست کنید

تست‌هایی که در اوایل کار انجام می‌شوند بسیار موثرتر از برنامه‌های تست عریض و طویلی هستند که گوشه‌ای خاک می‌خورند.

نکته ۶۳، تا زمانی که تست‌ها اجرا نشده باشند، برنامه‌نویسی به پایان نرسیده است

یعنی همین!

نکته ۶۴، از خرابکاران برای تست کردنِ تستِ خود استفاده کنید

برای این که مطمئن شوید ابزارِ تست شما درست کار می‌کند، ایرادهایی عمدی در برنامهٔ خود وارد کنید تا ببینید آیا کشف می‌شوند یا نه.

نکته ۶۵، خطوط برنامه را تست نکنید، حالت‌های مختلف برنامه را تست نمایید

حالت‌های مهمی که برنامه با آنها مُواجِه می‌شود را مشخص کرده و تست کنید. فقط تست کردنِ خطوط برنامه دردی را دوا نمی‌کند.

نکته ۶۶، ایرادها را فقط یک بار بیابید

وقتی که شخصِ تست‌کننده ایرادی را پیدا می‌کند، باید آخرین باری باشد که آن را می‌یابد. تست‌کننده‌های خودکار باید از این به بعد آن ایراد را پیگیری کنند.

نکته ۶۷، انگلیسی فقط یک زبان برنامه‌نویسی است

مستندات را همان طور بنویسید که برنامه‌ها را می‌نویسید. از MVC، تولیدکننده‌های خودکار و فایل‌های کمکی بهره گیرید.

نکته ۶۸، مستندات را روی هوا نسازید

مستنداتی که جداگانه از برنامه ایجاد شده‌اند بسیار نامحتمل است که به‌روز و صحیح باشند.

نکته ۶۹، به مرورِ زمان از انتظارات کاربران خود فراتر روید

سعی کنید انتظاراتِ کاربرانِ خود را درک کنید و فقط کافیست کمی بیشتر از آن چه که می‌خواستند انجام دهید.

نکته ۷۰، کارِ خود را امضا کنید

هنرمندانِ گذشته به امضای کارهای خود افتخار می‌کردند. شما هم باید برنامه‌ای بنویسید که به امضای خود در پای آن افتخار کنید.