# QA

## Khái niệm

* Phần mềm: Chương trình máy tính, thủ tục, các tài liệu liên quan(nếu có) và dữ liệu liên quan.
* Chất lượng phần mềm: Sự đáp ứng các yêu cầu chức năng, hiệu năng, các chuẩn (đặc tả) được phát triển, các đặc trưng mong muốn từ mọi phần mềm chuyên nghiệp.
* Đảm bảo chất lượng phần mềm: Một cách hệ thống, tật hợp các hành động được lập kế hoạch cần thiết để đem lại sự tin cậy là tiến trình phát triển phần mềm/tiến trình bảo trì tuân thủ theo các yêu cầu kỹ thuật về chức năng đã đưa ra cũng như yêu cầu quản lý về giữ lịch trình và hoạt động trong phạm vi ngân sách.
* Error, Fault, Failures
  + Error: Là các phần code sai do lỗi cú pháp, logic hoặc lỗi do phân tích thiết kế.
  + Fault: Là các error dẫn tới hoạt động không chính xác của phần mềm. Không phải error nào cùng gây ra fault.
  + Failures: Fault sẽ trở thành failure khi nó được kích hoạt. Một số đường chạy gây ra failures, một số không.
* V&V:
  + Verification(xác minh): Tiến trình đánh giá 1 system hoặc component xem sản phẩm của 1 pha phát triển đã cho có thỏa mãn điều kiện đưa ra ở đầu pha không.
  + Validation(xác nhận): Tiến trình đánh giá 1 system hoặc 1 component trong hoặc sau development process để xác định xem nó có thỏa mãn yêu cầu đã đặc tả hay không
  + IV&V: Independent Verification and Validation: Xác nhận và xác minh độc lập (Yêu cầu thực hiện việc đánh giá bởi người không phải lập trình viên, thường là chuyên gia trong lĩnh vực)
* MCCall: Là 1 bộ tiêu chí yêu cầu về chất lượng phần mềm. Có 11 tiêu chí được chia làm 3 nhóm:
  + Tiêu chí vận hành sản phẩm:
    - Chạy tốt như nào
    - Tính đúng đắn, tin cậy, hiệu quả, toàn vẹn, khả dụng.
  + Tiêu chí sửa đổi
    - Có thể thay đổi, kiểm thử, tái triển khai tốt như nào.
    - Tính bảo trì được, tính linh hoạt, tính kiểm thử được.
  + Tiêu chí chuyển giao sản phẩm
    - Có thể chuyển sang nền tảng khác và giao tiếp với các hệ thống khác tốt như nào.
    - Khả năng di động, khả năng tái sử dụng, khả năng tương thích.
* Review Inspection, Walkthrough (rà soát ngang hàng)
  + Inspection: Kỹ thuật đánh giá chính thức. Tài liệu, sản phẩm... được những người không phải là tác giả hoặc trực tiếp liên quan(Ngươi kiểm duyệt, tác giả, tester, thiết kế, coder) kiểm tra một cách chi tiết để phát hiện lỗi, các vi phạm tiêu chuẩn, hoặc các vấn đề khác (nếu có).
    - Về cơ bản, nó được tổ chức và thực hiện chặt chẽ hơn walkthrough. Vai trò của những người tham gia được phân định rõ ràng. Tài liệu chuẩn bị cho việc xem xét được chuẩn bị trước chu đáo.
    - Quá trình duyệt thảo bắt đầu sau giai đoạn code và unit test. Sau buổi họp các lỗi tìm đc sẽ đc sửa lại, rồi đem ra duyệt thảo lại cho đến khi đạt tiêu chuẩn mới kết thúc quá trình này
  + Walkthrough: Kỹ thuật đánh giá không chính thức(nên ko có ng quản lý, giám đốc dự án). Những người tham gia phải xem tài liệu trước cuộc họp (ít nhất vài ngày). Tác giả giải thích tài liệu/ sản phẩm đó cho nhóm (tác giả, điều phối viên, giám định viên, đại diện ng dùng, chuyên gia bảo trì).
    - Mọi người sẽ đặt câu hỏi hoặc cho ý kiến bổ sung về một số lĩnh vực để bảo đảm chất lượng kỹ thuật của tài liệu hoặc sản phẩm.
    - Buổi giám định có thể xảy ra vào bất kì lúc nào và bất kì đâu trong việc phát triển sản phẩm phần mềm. Mục đích chính của họp giám định chỉ là để tìm lỗi nhanh, ko tìm giải pháp. Sau giám định, tác giả của phải làm lại sửa mọi lỗi.
* Checkist cơ bản (GUI, web, code)

## Bài tập

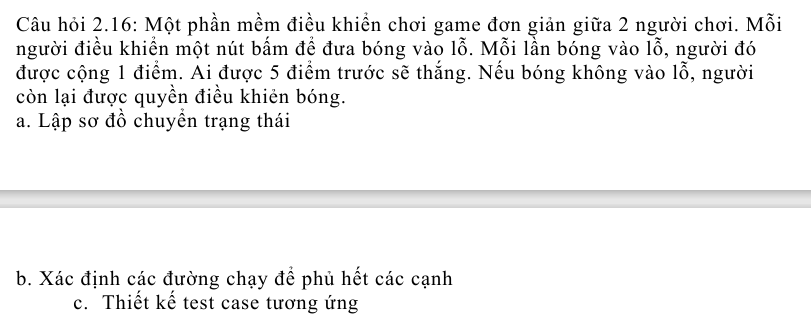
# TEST

## Khái niệm

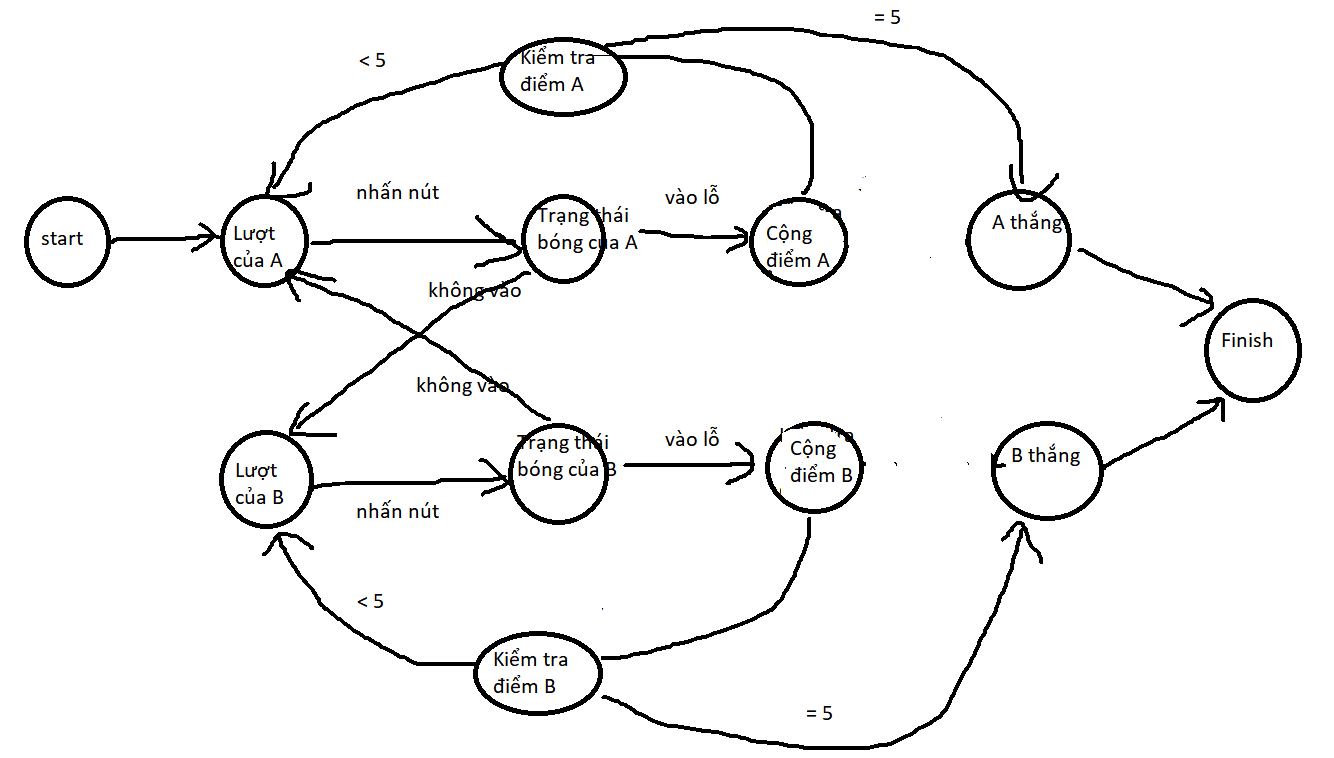
* Tại sao phải kiểm thử phần mềm? Mục tiêu kiểm thử là gì?
  + Kiểm thử phần mềm: Kiểm thử phần mềm là quá trình thực thi một hệ thống phần mềm để xác định xem phần mềm có đúng với đặc tả không và môi trường hoạt động có đúng yêu cầu không
  + Mục tiêu trực tiếp
    - Xác định và phát hiện nhiều lỗi nhất có thể trong phần mềm được kiểm thử
    - Sau khi sửa chữa các lỗi đã xác định và kiểm tra lại, làm cho phần mềm đã được kiểm thử đến một mức độ chấp nhận được về chất lượng.
    - Thực hiện các yêu cầu kiểm thử cần thiết một cách hiệu quả, trong phạm vi ngân sách và thời gian cho phép.
  + Mục tiêu gián tiếp: Biên dịch một bản ghi về các lỗi phần mềm để sử dụng trong công tác phòng chống lỗi
* Test cases:
  + Khái niệm: Ca kiểm thử là dữ liệu để kiểm tra hoạt động của chương trình.
  + Ca kiểm thử tốt: được thiết kế để phát hiện một lỗi của chương trình.
  + Kiểm thử thành công: phát hiện ra lỗi.
  + Mục đích:
    - Chứng minh được sự tồn tại của lỗi.
    - Không chứng minh được sự không có lỗi
* Level kiểm thử (Testing Level):
  + Unit Testing: Là hoạt động kiểm thử nhỏ nhất. Kiểm thử thực hiện trên các hàm hay thành phần riêng lẻ. Cần có hiểu biết về thiết kế chương trình và code. Thực hiện bởi Lập trình viên.
  + Integration Testing: Kiểm thử tích hợp nhằm phát hiện lỗi giao tiếp xảy ra giữa các thành phần cũng như lỗi của bản thân từng thành phần. Các thành phần có thể là các module, các ứng dụng riêng lẻ, các ứng dụng client/server trên cùng 1 mạng.
  + System Testing: Là 1 mức của tiến trình kiểm thử phần mềm khi các module và tích hợp các module đã được test. Mục đích để đánh giá phần mềm có tuân thủ theo các yêu cầu đã đưa ra không.
  + Acceptance Testing: Là kiểm thử bên phía người dùng. Được chi ra làm 2 loại:
    - Alpha Test, người dùng kiểm thử phần mềm ngay tại nơi phát triền phần mềm, lập trình viên sẽ ghi nhận các lỗi hoặc phản hồi, và lên kế hoạch sửa chữa.
    - Beta Test, phần mềm được gửi tới cho người dùng để kiểm thử ngay trong môi trường thực, lỗi hoặc phản hồi cũng sẽ được gửi ngược lại cho lập trình viên để sửa chữa.
* Quy trình kiểm thử:

## Bài tập

### 2.16

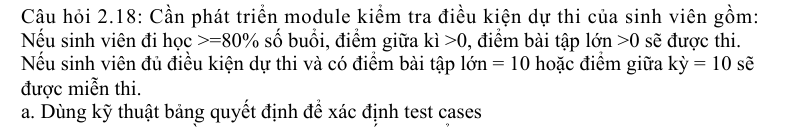


1. Sơ đồ chuyển trạng thái



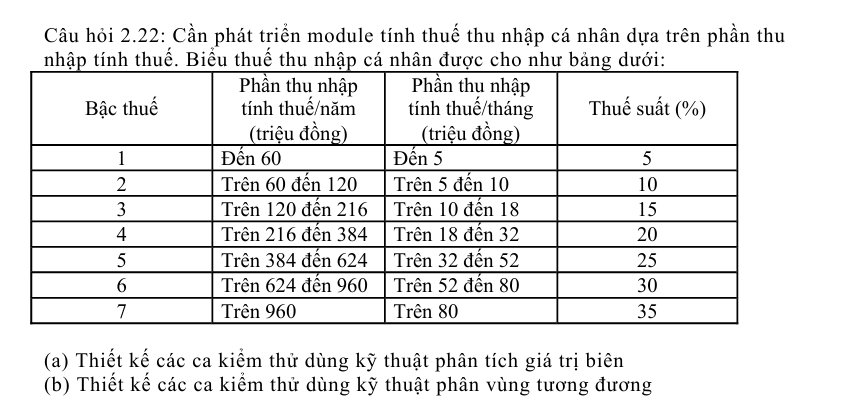
1. Đường chạy
2. Test case

### 2.18



|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Rule1 | Rule2 | Rule3 | Rule4 | Rule 5 | Rule 6 | Rule 7 | Rule 8 | Rule 9 | Rule 10 | Rule 11 |
| Đi học >= 80% số buổi | Yes | Yes | Yes | Yes | Yes | Yes | Yes | No | No | No | No |
| Điểm giữa kì>0 | Yes | Yes | Yes | Yes | Yes | No | No | Yes | Yes | No | No |
| Điểm BTL>0 | Yes | Yes | Yes | Yes | No | Yes | No | Yes | No | Yes | No |
| Điểm giữa kì=10 | Yes | Yes | No | No | No | No | No | No | No | No | No |
| Điểm BTL=10 | Yes | No | Yes | No | No | No | No | No | No | No | No |
| Kết quả | miễn thi | miễn thi | miễn thi | được thi | không được thi | không được thi | không được thi | không được thi | không được thi | không được thi | không được thi |

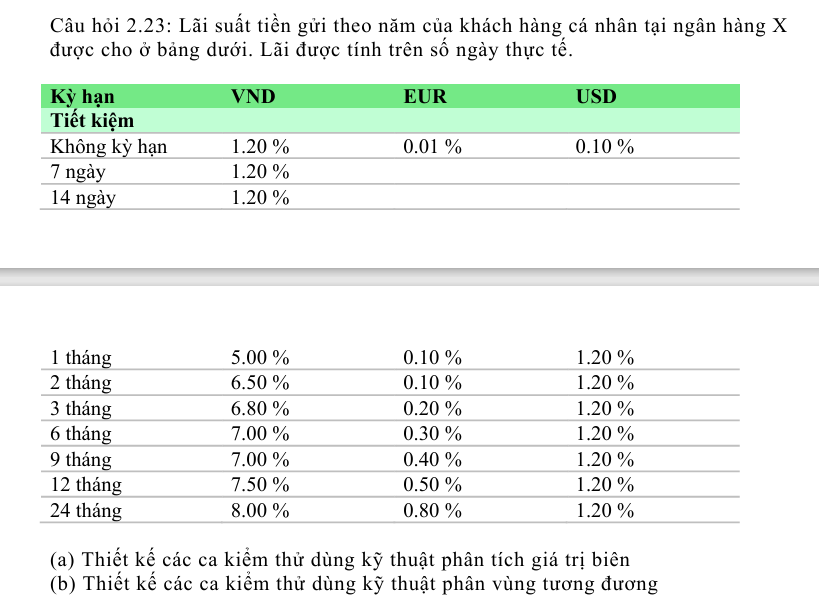
### 2.22







### 2.23



1. Phân tích giá trị biên

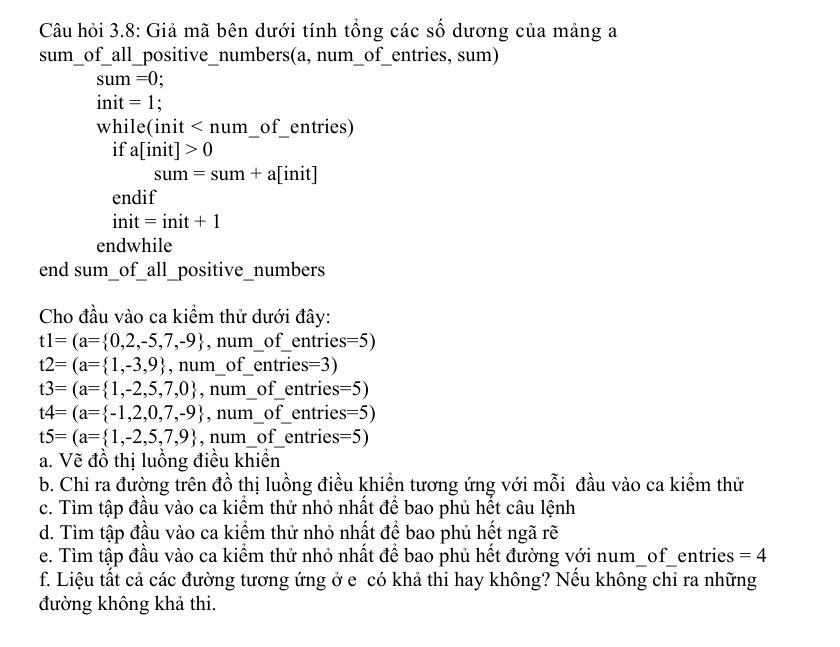
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Kỳ hạn | Giá trị (ngày) | Kết quả mong đợi (VND) % | Kết quả mong đợi (EUR) % | Kết quả mong đợi (USD) % |
| Không kỳ hạn | 0 | 0 | 0 | 0 |
| Không kỳ hạn | 365 | 1.2 | 0.01 | 0.1 |
| 7 ngày |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

2.24: Một chương trình phân loại tam giác đọc vào các giá trị số nguyên trong khoảng [0,100]. 3 giá trị A, B, và C được dùng để biểu diễn độ dài 3 cạnh tam giác. Chương trình in ra thông báo 3 giá trị này có phải là 3 cạnh tam giác không.

(a) Thiết kế các ca kiểm thử dùng kỹ thuật phân tích giá trị biên

(b) Thiết kế các ca kiểm thử dùng kỹ thuật phân vùng tương đương

### 3.8



1. Vẽ luồng đồ thị điều khiển

sum\_of\_all\_positive\_numbers(a, num\_of\_entries, sum)

sum =0; // 1

init = 1; // 2

while(init <= num\_of\_entries) // 3

if a[init] > 0 // 4

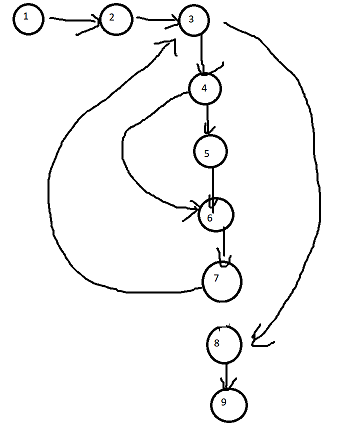
sum = sum + a[init] // 5

endif // 6

init = init + 1 // 7

endwhile //8

end sum\_of\_all\_positive\_numbers // 9

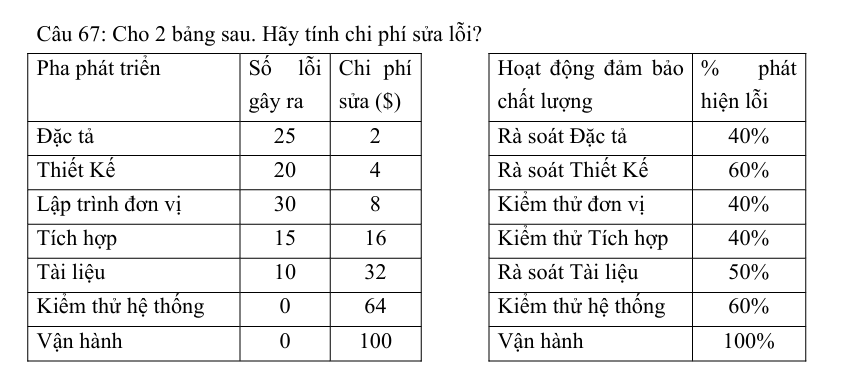


1. Chỉ ra các đường

|  |  |  |
| --- | --- | --- |
| Tên luồng | Đầu vào | Đường đi |
| t1 | a={0,2,-5,7,-9}, num\_of\_entries=5 | 1,2 ,3,4,6,7 ,3,4,5,6,7 ,3,4,6,7 3,4,5,6,7, 3,4,6,7 ,3,8,9 |
| t2 | a={1,-3,9}, num\_of\_entries=3 | 1,2 ,3,4,5,6,7 ,3,4,6,7 3,4,5,6,7, 3,8,9 |
| t3 | a={1,-2,5,7,0}, num\_of\_entries=5 | 1,2 ,3,4,6,7 ,3,4,5,6,7 ,3,4,5,6,7 ,3,4,6,7 ,3,8,9 |
| t4 | a={-1,2,0,7,-9}, num\_of\_entries=5 | 1,2 ,3,4,6,7 ,3,4,5,6,7 ,3,4,6,7 ,3,4,5,6,7 ,3,4,6,7 ,3,8,9 |
| t5 | a={1,-2,5,7,9}, num\_of\_entries=5 | 1,2 ,3,4,5,6,7,3,4,6,7 ,3,4,5,6,7,3,4,5,6,7,3,4,5,6,7 ,3,8,9 |

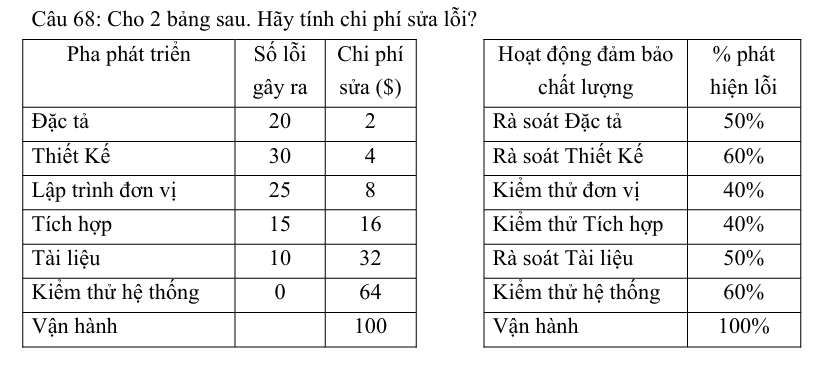
1. Tìm tập đầu vào ca kiểm thử nhỏ nhất để bao phủ hết câu lệnh
2. Tìm tập đầu vào ca kiểm thử nhỏ nhất để bao phủ hết ngã rẽ
3. Tìm tập đầu vào ca kiểm thử nhỏ nhất để bao phủ hết đường với num\_of\_entries = 4

### 67. Chi phí 1



|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Pha phát triển | Số lỗi gây ra | Tổng số lỗi | %Phát hiện lỗi | Số lỗi phát hiện | Số lỗi còn lại | Chi phí sửa | Tổng chi phí |
|  |  |  |  |  |  |  |  |
| Đặc tả | 25 | 25 | 40% | 10 | 15 | 2 | 20 |
| Thiết kế | 20 | 35 | 60% | 21 | 14 | 4 | 84 |
| Lập trình đơn vị | 30 | 44 | 40% | 17.6 | 26.4 | 8 | 140.8 |
| Tích hợp | 15 | 41.4 | 40% | 16.56 | 24.84 | 16 | 264.96 |
| Tài liệu | 10 | 34.84 | 50% | 17.42 | 17.42 | 32 | 557.44 |
| Kiểm thử hệ thống | 0 | 17.42 | 60% | 10.452 | 6.968 | 64 | 668.928 |
| Vận hành | 0 | 6.968 | 100% | 6.968 | 0 | 100 | 696.8 |
| Tổng |  |  |  |  |  |  | **2432.928** |

### 68. Chi phí 2



|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Pha phát triển | Số lỗi gây ra | Tổng số lỗi | % Phát hiện lỗi | Số lỗi phát hiện | Số lỗi còn lại | Chi phí sửa | Tổng chi phí |
| Đặc tả | 20 | 20 | 50% | 10 | 10 | 2 | 20 |
| Thiết kế | 30 | 40 | 60% | 24 | 16 | 4 | 96 |
| Lập trình đơn vị | 25 | 41 | 40% | 16.4 | 24.6 | 8 | 131.2 |
| Tích hợp | 15 | 39.6 | 40% | 15.84 | 23.76 | 16 | 253.44 |
| Tài liệu | 10 | 33.76 | 50% | 16.88 | 16.88 | 32 | 540.16 |
| Kiểm thử hệ thống | 0 | 16.88 | 60% | 10.128 | 6.752 | 64 | 648.192 |
| Vận hành | 0 | 6.752 | 100% | 6.752 | 0 | 100 | 675.2 |
| **Tổng** |  |  |  |  |  |  | **2364.192** |