1. ***Application: BINARY HEAP của trang web*** [***https://visualgo.net/en***](https://visualgo.net/en)***:***

Bước 1: Vào trang <https://visualgo.net/en>.

Bước 2: Để truy cập vào chức năng Hash Table, người dùng chọn mục được mô tả như hình bên dưới:



1. ***Environment:***

Bài kiểm thử được thực hiện trên hai trình duyệt cụ thể là:

* Google chrome.
* Firefox.

1. ***Các thuật toán kiểm thử Black-Box Testing được sử dụng trong bài:***

* Boundary value analysis technique.
* Equivalence class partitioning technique.
* Decision table technique.
* Use-case testing.

1. ***Non-function:***

* **Test scenario 1:** Khởi tạo Binary Heap từ 1 đến 18 phần tử

Mục đích: Kiểm tra số lượng phần tử có thể khởi tạo cho Binary Heap

Quy trình thực hiện:

* + Bước 1: Chọn Create(A) - O(N log N) hoặc Create(A) - O(N)
  + Bước 2: Nhập số
  + Bước 3: Nhấn Go

Các tình huống cần test:

* + Nhập số phần tử trong khoảng (1, 18)
  + Nhập một phần tử.
  + Nhập 18 phần tử.
  + Không nhập phần tử nào.
  + Nhập lớn hơn 18 phần tử
* **Test scenario 2:** Kiểm tra giới hạn giá trị của 1 phần tử khi khởi tạo Binary Heap

Mục đích: Kiểm tra giới hạn giá trị của 1 phần tử khi khởi tạo Binary Heap

Quy trình thực hiện:

* + Bước 1: Chọn Create(A) - O(N log N) hoặc Create(A) - O(N)
  + Bước 2: Nhập số
  + Bước 3: Nhấn Go

Các tình huống cần test:

* + Nhập số phần tử trong khoảng (-100, 100)
  + Nhập số phần tử ngoài khoảng (-100, 100)
* **Test scenario 3:** Kiểm các giá trị giống nhau và giá trị đặc biệt khi khởi tạo Binary Heap

Mục đích: Kiểm tra các giá trị hợp lệ khi khởi tạo Binary Heap

Quy trình thực hiện:

* + Bước 1: Chọn Create(A) - O(N log N) hoặc Create(A) - O(N)
  + Bước 2: Nhập số
  + Bước 3: Nhấn Go

Các tình huống cần test:

* + Nhập phần tử có ký tự đặc biệt
  + Nhập phần tử là là số không nguyên
  + Nhập các phần tử có giá trị trùng nhau
* **Test scenario 4:** Insert phần tử thứ 2 đến 18 trong Binary Heap

Mục đích: Kiểm tra số lượng phần tử có thể thêm cho Binary Heap

Quy trình thực hiện:

* + Bước 1: Chọn Insert(v)
  + Bước 2: Nhập số
  + Bước 3: Nhấn Go

Các tình huống cần test:

* + Thêm phần tử thứ 2
  + Thêm phần tử thứ 3 đến 17
  + Thêm phần tử thứ 18
  + Thêm phần tử thứ 19
* **Test scenario 5:** Kiểm các giá trị giống nhau và giá trị đặc biệt khi thêm giá trị cho Binary Heap

Mục đích: Kiểm tra các giá trị hợp lệ khi thêm giá trị cho Binary Heap

Quy trình thực hiện:

* + Bước 1: Chọn Insert(v)
  + Bước 2: Nhập số
  + Bước 3: Nhấn Go

Các tình huống cần test:

* + Thêm phần tử có ký tự đặc biệt
  + Thêm phần tử là là số không nguyên
  + Thêm phần tử có giá trị đã có trong Binary Heap
* **Test scenario 6:** Cập nhật các phần tử từ vị trí 1 đến vị trí cuối trong Binary Heap

Mục đích: Kiểm tra các vị trí có thể cập nhật cho Binary Heap

Quy trình thực hiện:

* + Bước 1: UpdateKey(i, newv)
  + Bước 2: Nhập số
  + Bước 3: Nhấn Go

Các tình huống cần test:

* + Cập nhật phần tử ở vị trí 1
  + Cập nhật phần tử ở vị trí 2 đến vị trí kế cuối
  + Cập nhật phần tử ở vị trí cuối cùng
  + Cập nhật phần tử ở vị trí không tồn tại
* **Test scenario 7:** Cập nhật các giá trị từ 1 đến 99 cho Binary Heap

Mục đích: Kiểm tra các giá trị có thể cập nhật cho Binary Heap

Quy trình thực hiện:

* + Bước 1: UpdateKey(i, newv)
  + Bước 2: Nhập số
  + Bước 3: Nhấn Go

Các tình huống cần test:

* + Cập nhật phần tử có giá trị 0
  + Cập nhật phần tử có giá trị từ 1 đến 99
  + Cập nhật phần tử có giá trị lớn hơn 99
  + Cập nhật phần tử có giá trị rỗng
* **Test scenario 8:** Kiểm các giá trị giống nhau và giá trị đặc biệt khi thêm giá trị cho Binary Heap

Mục đích: Kiểm tra các giá trị hợp lệ khi cập nhật cho Binary Heap

Quy trình thực hiện:

* + Bước 1: UpdateKey(i, newv)
  + Bước 2: Nhập số
  + Bước 3: Nhấn Go

Các tình huống cần test:

* + Cập nhật phần tử có giá trị không phải số nguyên
  + Cập nhật phần tử có giá trị đã tồn tại trong Binary Heap
* **Test scenario 9:** Delete từ vị trí 1 đến vị trí cuối trong Binary Heap

Mục đích: Kiểm tra các vị trí có thể xóa trong Binary Heap

Quy trình thực hiện:

* + Bước 1: Delete(i)
  + Bước 2: Nhập số
  + Bước 3: Nhấn Go

Các tình huống cần test:

* + Xóa phần tử ở vị trí 1
  + Xóa phần tử ở vị trí 2 đến vị trí kế cuối
  + Xóa phần tử ở vị trí cuối cùng
  + Xóa phần tử ở vị trí nằm ngoài kích thước Binary Heap
* **Test scenario 10:** Delete các vị trí đặc biệt trong Binary Heap

Mục đích: Kiểm tra các vị trí hợp lệ có thể xóa trong Binary Heap

Quy trình thực hiện:

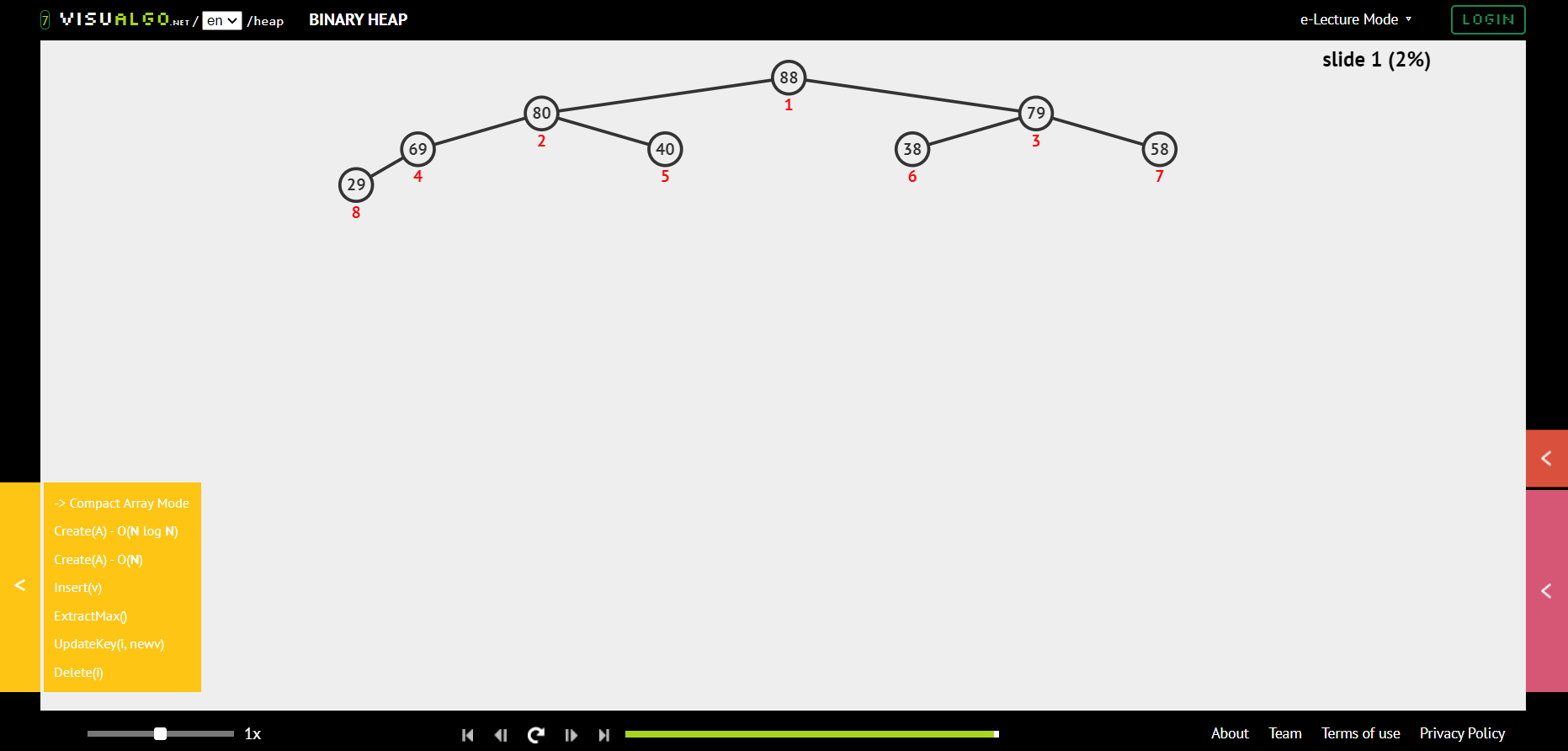
* + Bước 1: Delete(i)
  + Bước 2: Nhập số
  + Bước 3: Nhấn Go

Các tình huống cần test:

* + Xóa phần tử ở vị trí rỗng
  + Xóa phần tử ở vị trí không phải số nguyên

5. ***Function:***

* **Test scanario**: Các chức năng Create, Insert, Update, Delete trong Binary Heap

Mục đích: Kiểm tra các chức năng Create, Insert, Update, Delete trong Binary Heap 

1. Create

Quy trình thực hiện:

Bước 1: Chọn Create.

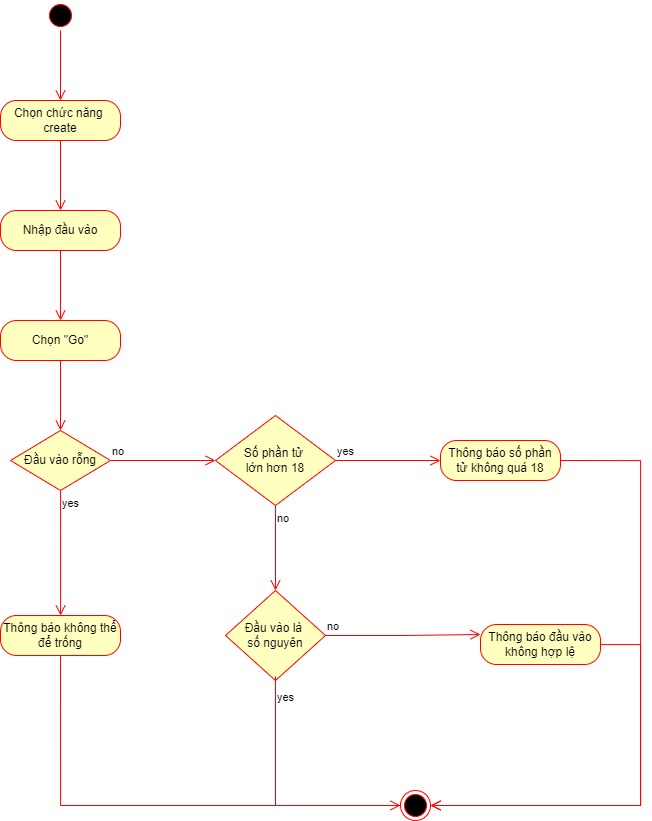
Bước 2: Nhập số.

Bước 3: Chọn Go.

Đầu vào:

* Số trong đoạn [-100 .. 100].
* Số ngoài đoạn [-100 .. 100].
* Không nhập gì cả.
* Nhập vào các ký tự đặc biệt.
* Nhập vào số phần tử trong đoạn  1≤x≤18.
* Nhập vào số phần tử lớn hơn 18.
* Nhập vào các số không nguyên.
* Nhập vào các số trùng nhau.

Activity Graph:



Decision table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Rule 1-4 | Rule 5-6 | Rule 7 | Rule 8 |
| Condition: |  |  |  |  |
| Đầu vào rỗng | T | F | F | F |
| Đầu vào lớn hơn 18 phần tử | - | T | F | F |
| Đầu vào là các số nguyên | - | - | T | F |
| Action: |  |  |  |  |
| Thực hiện create |  |  | X |  |
| Thông báo lỗi | X | X |  | X |

Các tình huống cần test dựa vào Decision table:

- Không nhập đầu vào.

- Nhập đầu vào có số phần tử lớn hơn 18.

- Nhập vào số phần tử trong đoạn [1, 18] và trong các số nhập vào có chứa ít nhất 1 phần tử không phải số nguyên.

- Nhập vào số phần tử trong đoạn [1, 18] và các số nhập vào đều là số nguyên.

1. Insert

Quy trình thực hiện:

Bước 1: Chọn Insert.

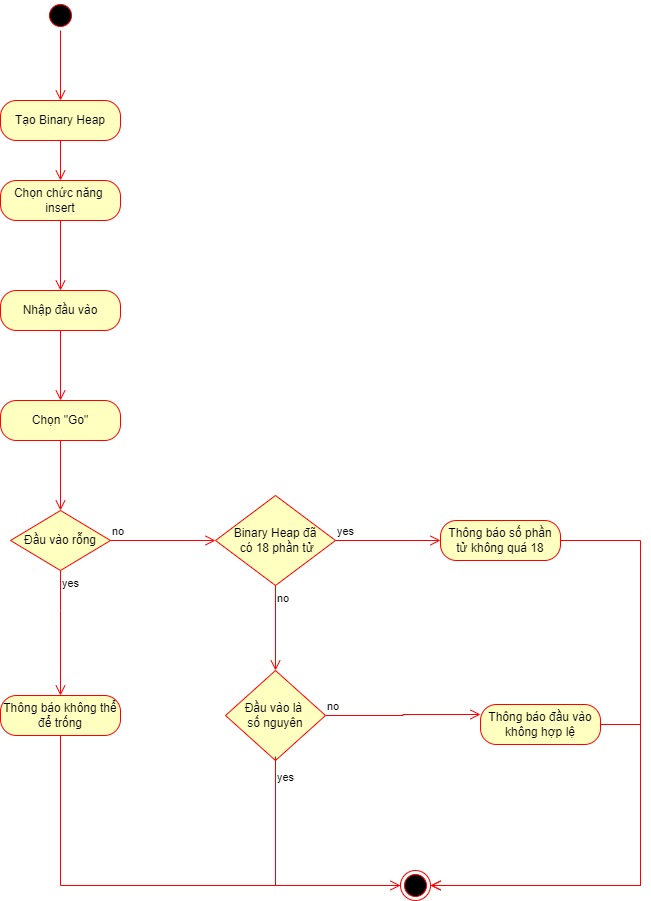
Bước 2: Nhập số.

Bước 3: Chọn Go.

Đầu vào:

* Số trong đoạn [-100 .. 100].
* Số ngoài đoạn [-100 .. 100].
* Không nhập gì cả.
* Nhập vào các ký tự đặc biệt.
* Chèn phần tử có vị trí trong đoạn  1≤x≤18.
* Chèn phần tử có vị trí lớn hơn 18.
* Nhập vào các số không nguyên.
* Nhập vào số đã tồn tại.

Activity Graph:



Decision table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Rule 1-4 | Rule 5-6 | Rule 7 | Rule 8 |
| Condition: |  |  |  |  |
| Đầu vào rỗng | T | F | F | F |
| Binary Heap đã có 18 phần tử | - | T | F | F |
| Đầu vào là số nguyên | - | - | T | F |
| Action: |  |  |  |  |
| Thực hiện insert |  |  | X |  |
| Thông báo lỗi | X | X |  | X |

Các tình huống cần test dựa vào Decision table:

- Không nhập đầu vào.

- Insert vào Binary Heap có 18 phần tử.

- Insert vào Binary Heap có [1,17] phần tử và đầu vào là số nguyên.

- Insert vào Binary Heap có [1,17] phần tử và đầu vào không phải số nguyên.

1. Update

Quy trình thực hiện:

Bước 1: Chọn UpdateKey.

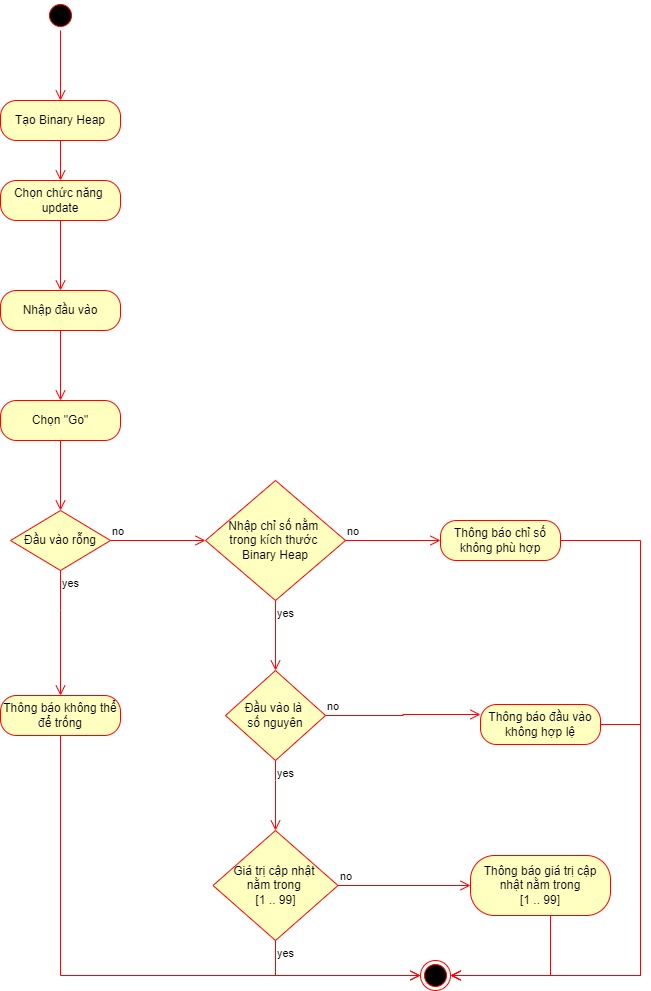
Bước 2: Nhập số.

Bước 3: Chọn Go.

Đầu vào:

* Số trong đoạn [1 .. 99].
* Số ngoài đoạn [1 .. 99].
* Không nhập gì cả.
* Cập nhật phần tử từ vị trí 1 đến vị trí cuối.
* Cập nhật phần tử có vị trí lớn hơn vị trí cuối.
* Nhập vào các số không nguyên.
* Nhập vào số đã tồn tại.

Activity Graph:



Decision table:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Rule 1-8 | Rule 9-12 | Rule 13-14 | Rule 15 | Rule 16 |
| Condition: |  |  |  |  |  |
| Đầu vào rỗng | T | F | F | F | F |
| Đầu vào chỉ số nằm trong kích thước Binary Heap | - | F | T | T | T |
| Đầu vào là số nguyên | - | - | F | T | T |
| Đầu vào giá trị cập nhật nằm trong [1 .. 99] | - | - | - | T | F |
| Action: |  |  |  |  |  |
| Thực hiện update |  |  |  | X |  |
| Thông báo lỗi | X | X | X |  | X |

Các tình huống cần test dựa vào Decision table:

- Không nhập đầu vào.

- Nhập chỉ số đầu vào lớn hơn kích thước Binary Heap.

- Nhập chỉ số đầu vào không lớn hơn kích thước Binary Heap và đầu vào không phải số nguyên.

- Nhập chỉ số đầu vào không lớn hơn kích thước Binary Heap và đầu vào là số nguyên nằm trong khoảng [1 .. 99].

- Nhập chỉ số đầu vào không lớn hơn kích thước Binary Heap và đầu vào là số nguyên nằm ngoài khoảng [1 .. 99].

1. Delete

Quy trình thực hiện:

Bước 1: Chọn Delete.

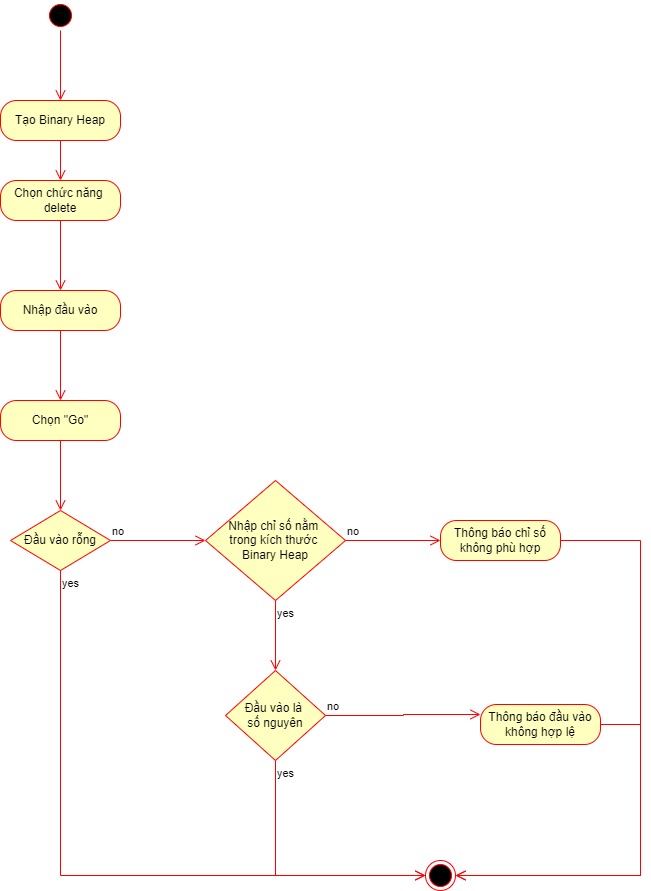
Bước 2: Nhập số.

Bước 3: Chọn Go.

Đầu vào:

* Không nhập gì cả.
* Xóa phần tử từ vị trí 1 đến vị trí cuối.
* Xóa phần tử có vị trí lớn hơn vị trí cuối.
* Nhập vào các số không nguyên.

Activity Graph:



.

Decision table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Rule 1-4 | Rule 5-6 | Rule 7 | Rule 8 |
| Condition: |  |  |  |  |
| Đầu vào rỗng | T | F | F | F |
| Đầu vào chỉ số nằm trong kích thước Binary Heap | - | F | T | T |
| Đầu vào là các số nguyên | - | - | T | F |
| Action: |  |  |  |  |
| Thực hiện delete |  |  | X |  |
| Thông báo lỗi | X | X |  | X |

Các tình huống cần test dựa vào Decision table:

- Không nhập đầu vào.

- Nhập chỉ số đầu vào lớn hơn kích thước Binary Heap.

- Nhập chỉ số đầu vào không lớn hơn kích thước Binary Heap và đầu vào là số nguyên.

- Nhập chỉ số đầu vào không lớn hơn kích thước Binary Heap và đầu vào không phải số nguyên.