# What does the push_heap function in C++ do?

I wonder what does the push_heap function which takes three parameters do?

```
#include <iostream>
#include <cassert>
#include <algorithm>
#include <vector>
using namespace std;

class HeapCompare_f
    {
        public:

            bool operator() ( int x, int y ) const
            {
                return x > y;
            }
    };
int main()
{
    vector<int> vector1(5);

    for (int i = 0; i < 5; ++i)
        vector1[i] = 5-i;

    for (int i = 0; i < 5; ++i)
        cout << vector1[i];
    cout << endl;

    push_heap(vector1.begin(), vector1.end(),HeapCompare_f());

    for (int i = 0; i < 5; ++i)
        cout << vector1[i];
    cout << endl;



  return 0;
}
```

The output of this code is

```
54321
15324
```

Also I wonder how can I implement that function in C ? Because I will use it in A* algorithm which I am writing in C

c++ stl heap

asked Oct 26 '11 at 15:32

Alptugay
**634** 3 10 22

This code isn't correct, since it violates the precondition for `push_heap` that the given range is a valid heap except for the last element. – interjay Oct 26 '11 at 15:56

Seriously? en.cppreference.com/w/cpp/algorithm/push_heap – user405725 Oct 26 '11 at 15:57

## 2 Answers

This function does **not** turn a range of values into a heap!

```
std::push_heap(first, last [, comp])
```

assumes that the range `[first,last-1)` is already a valid heap and pushes the value at position `last-1` into the heap, moving it to the correct position to keep the heap-requirement valid. It uses either the `<` operator to determine the ordering of the elements or a user-specified comparator.

edited Oct 26 '11 at 16:11

answered Oct 26 '11 at 15:55

Christian Rau
**32.3k** 5 51 111

When I insert `make_heap(vector1.begin(), vector1.end());` the code before `push_heap` function, the output does not change. And you said that `push_heap` function pushes the value at `last-1` to the correct position. What is the correct position? In our case the value at `last-1` is `1`. Where should it be? At the beginning of the vector? At the end? If it is at the begginning as the output is `15324` why did the other elements' order changed? Shouldnt the output be `15432` as it moves the element at `last -1` to the begginning – Alptugay Oct 26 '11 at 17:24

@Alptugay The correct position is any position for which the heap-requirement holds. If you don't understand this, then first look up what a heap data structure is (maybe the above link helps with that). `push_heap` only works if the range between `first` and `last-1` is already a heap. Othwerwise the results are undefined. And yes, `push_heap` can change the order of the elements in order to place the item at the correct position. But of course calling `make_heap` and then `push_heap` on the same range doesn't make a difference, as `last-1` is already at the correct position. – Christian Rau Oct 26 '11 at 17:34

You are using push_heap incorrectly.

After initializing your vector, you need to put it in heap order:

```
std::make_heap(vector1.begin(), vector1.end());
```

To add further elements into the heap, you need to first push each to the back of the vector, then call push_heap:

```
vector1.push_back(42);
std::push_heap(vector1.begin(), vector1.end());
```

Finally, to remove the first element in the heap, you need to call pop_heap, followed by popping the last element from the vector:

```
std::pop_heap(vector1.begin(), vector1.end());
vector1.pop_back();
```

The three-parameter heap functions let you specify a compare method to control the heap order, which you are doing correctly.

The reason for the manual push_back and pop_back calls is that the heap functions only see iterators into a container, and do not have access to the container itself. Since iterators are not sufficient to modify the contents of a container, this must be done manually by the owner of the container (you).

To avoid having to deal with any of this yourself, I'd recommend using a `std::priority_queue`.

edited Oct 26 '11 at 16:12

Christian Rau
**32.3k** 5 51 111

answered Oct 26 '11 at 16:05



zennehoy
**3,530** 7 24

Thank you for the info. You said that The three-parameter heap functions let you specify a compare method to control the heap order, but how does it do it? I actually dont see any order in the output. 15324 seems pretty unordered – Alptugay Oct 26 '11 at 17:13

@Alptugay It doesn't need to be ordered but it has to have the heap-requirement. And again: If you don't know what this means, look up what a heap data structure is. – Christian Rau Oct 26 '11 at 21:07