

PRIORITY QUEUE COMPARISON CLASSES

Problem Solving with Computers-II

C++

```
#include <iostream>
using namespace std;

int main(){
    cout<<"Hola Facebook!n";
    return 0;
}
```



From last class....

```
int main(){
    int arr[]={10, 2, 80};
    priority_queue<int> pq;
    for(int i=0; i < 3; i++)
        pq.push(arr[i]);

    while(!pq.empty()){
        cout<<pq.top()<<endl;
        pq.pop();
    }
    return 0;
}
```

What is the output of this code?

A. 10 2 80

B. 2 10 80

C. 80 10 2

D. 80 2 10

E. None of the above

std::priority_queue template arguments

The template for priority_queue takes 3 arguments:

```
template <
    class T,
    class Container= vector<T>,
    class Compare = less <T>
> class priority_queue;
```

- The first is the type of the elements contained in the queue.
- If it is the only template argument used, the remaining 2 get their default values:
 - a **vector<T>** is used as the internal store for the queue,
 - **less is a comparison** class that provides priority comparisons

Comparison class

- A class used to perform comparisons.
- Implements a function operator that compares two keys

```
class cmp{
    bool operator()(int& a, int& b) const {
        return a > b;
    }
};
```

//Use cmp to compare any two keys

```
Cmp foo;
cout<<foo(x, y);
```

Configure PQ with a comparison class

```
class cmp{
    bool operator()(int& a, int& b) const {
        return a > b;
    }
};
int main(){
    int arr[]={10, 2, 80};
    priority_queue<int, vector<int>, cmp> pq;
    for(int i=0; i < 3; i++)
        pq.push(arr[i]);

    while(!pq.empty()){
        cout<<pq.top()<<endl;
        pq.pop();
    }
    return 0;
}
```

What is the output of this code?

A. 10 2 80

B. 2 10 80

C. 80 10 2

D. 80 2 10

E. None of the above

Practice functors and PQs:

```
int main(){
    int arr[]={10, 2, 80};
    priority_queue<int*> pq;
    for(int i=0; i < 3; i++)
        pq.push(arr+i);

    while(!pq.empty()){
        cout<<*pq.top()<<endl;
        pq.pop();
    }
    return 0;
}
```

What is the output of this code?

A. 10 2 80

B. 2 10 80

C. 80 10 2

D. 80 2 10

E. None of the above

Sort array elements using a pq storing pointers

```
int main(){
    int arr[]={10, 2, 80};
    priority_queue<int*> pq;
    for(int i=0; i < 3; i++)
        pq.push(arr+i);

    while(!pq.empty()){
        cout<<*pq.top()<<endl;
        pq.pop();
    }
    return 0;
}
```

How can we change the way pq prioritizes pointers?

Write a comparison class to print the integers in the array in sorted order

```
int main(){
    int arr[]={10, 2, 80};
    priority_queue<int*, vector<int*>, cmpPtr> pq;
    for(int i=0; i < 3; i++)
        pq.push(arr+i);

    while(!pq.empty()){
        cout<<*pq.top()<<endl;
        pq.pop();
    }
    return 0;
}
```


Small group exercise

Write a ADT called `minStack` that provides the following methods

- `push()` // inserts an element to the “top” of the `minStack`
- `pop()` // removes the last element that was pushed on the stack
- `top ()` // returns the last element that was pushed on the stack
- `min()` // returns the minimum value of the elements stored so far

