**Assignment - 42 A Job Ready Bootcamp in C++, DSA and IOT**

**Queue**

1. Print all elements of a queue in C++ STL

Sol – 1.

#include<bits/stdc++.h>

using namespace std;

int main()

{

queue<int>q;

q.push(5);

q.push(4);

q.push(3);

q.push(2);

q.push(1);

while(!q.empty())

{

cout<<q.front()<<" ";

q.pop();

}

cout<<endl;

return 0;

}

1. Create a railways ticket counter and allot ticket to first come first take using c++ STL

Sol – 2.

#include<iostream>

#include<queue>

#include<string>

#include<math.h>

#include<time.h>

using namespace std;

class ticket

{

public :

string passenger;

int tnumber;

};

ticket ticketcounter()

{

ticket temp;

cout<<"Enter your name : ";

cin>>temp.passenger;

srand(time(NULL));

temp.tnumber=rand();

return temp;

}

void showticket(queue<ticket>q)

{

ticket temp;

cout<<"| Ticket Number | Passenger Name "<<endl;

cout<<"-----------------------------------------"<<endl;

while(!q.empty())

{

temp=q.front();

cout<<"| "<<temp.tnumber<<" | "<<temp.passenger<<endl;

q.pop();

}

cout<<"-----------------------------------------"<<endl;

}

int main()

{

queue<ticket>q;

int size;

cout<<"Enter the no. of passenger : ";

cin>>size;

for(int i=0;i<size;i++)

{

q.push(ticketcounter());

}

showticket(q);

return 0;

}

1. Check if a queue is empty or not using queue::size() function

Sol – 3.

#include<bits/stdc++.h>

using namespace std;

int main()

{

queue<int>q;

q.push(5);

q.push(4);

if(q.size()==0)

{

cout<<"Queue is empty"<<endl;

}

else

cout<<"Queue is not empty"<<endl;

q.pop();

q.pop();

if(q.size()==0)

{

cout<<"Queue is empty"<<endl;

}

else

cout<<"Queue is not empty"<<endl;

return 0;

}

1. Manage a queue for multiple input and store in ascending order of his first character

Sol – 4.

#include<bits/stdc++.h>

using namespace std;

int main()

{

int c=1;

queue<string>q;

string str;

while(true)

{

cout<<"Press 1 to enter more or 0 to stop : ";

cin>>c;

if(c==0)

break;

cout<<"Enter a single word string : ";

cin>>str;

q.push(str);

}

int size=q.size();

string temp[size];

for(int i=0;i<size;i++)

{

temp[i]=q.front();

q.pop();

}

sort(temp,temp+size);

for(int i=0;i<size;i++)

{

q.push(temp[i]);

}

while(!q.empty())

{

cout<<q.front()<<endl;

q.pop();

}

return 0;

}

1. Schedule a interview by using applicant's reaching time using c++ STL

Sol – 5.

#include<bits/stdc++.h>

using namespace std;

struct Applicant

{

string name;

string currenttime;

};

Applicant getEntry()

{

Applicant temp;

cout<<"Enter your name : ";

cin>>temp.name;

cout<<"Enter current time : ";

cin>>temp.currenttime;

return temp;

}

int main()

{

queue<Applicant>l1;

for(int i=0;i<5;i++)

{

l1.push(getEntry());

}

Applicant temp;

for(int i=0;i<5;i++)

{

temp=l1.front();

cout<<"Applicant Name : "<<temp.name<<endl<<"Your Interview time is : "<<temp.currenttime<<endl;

l1.pop();

}

return 0;

}

6. In C++ STL, Queue is a type of container that follows FIFO (First-in-First-Out)

elements arrangement i.e. the elements which insert first will be removed first. In the

queue, elements are inserted at one end known as "back" and are deleted from

another end known as "front".empty() and size() function of the queue with the

Example.

Sol – 6.

#include<bits/stdc++.h>

using namespace std;

int main()

{

queue<int>q;

q.push(5);

q.push(4);

q.push(3);

q.push(2);

q.push(1);

cout<<"First element is : "<<q.front()<<endl;

cout<<"Last element is : "<<q.back()<<endl;

cout<<"Removing 2 elements..."<<endl;

q.pop();

q.pop();

cout<<"Queue is"<<endl;

while(!q.empty())

{

cout<<q.front()<<" ";

q.pop();

}

cout<<endl;

return 0;

}

7. Exchange the contents of two queues but the queues must be of the same data type,

although sizes may differ.

Sol – 7.

#include<bits/stdc++.h>

using namespace std;

int main()

{

queue<int>q;

q.push(5);

q.push(4);

q.push(3);

q.push(2);

q.push(1);

queue<int>q2;

for(int i=6;i<10;i++)

q2.push(i);

q.swap(q2);

cout<<"Queue is"<<endl;

while(!q.empty())

{

cout<<q.front()<<" ";

q.pop();

}

cout<<endl;

cout<<"Queue 2 is"<<endl;

while(!q2.empty())

{

cout<<q2.front()<<" ";

q2.pop();

}

cout<<endl;

return 0;

}

8. Insert a new element into the queue container, the new element is added to the end

of the queue

Sol – 8.

#include<bits/stdc++.h>

using namespace std;

int main()

{

queue<int>q;

q.push(5);

q.push(4);

q.push(3);

cout<<"Queue is"<<endl;

while(!q.empty())

{

cout<<q.front()<<" ";

q.pop();

}

cout<<endl;

return 0;

}

9. Adds the element ‘g’ at the end of the queue.

Sol – 9.

#include<bits/stdc++.h>

using namespace std;

int main()

{

queue<char>q;

q.push('u');

q.push('o');

q.push('g');

cout<<"Queue is"<<endl;

while(!q.empty())

{

cout<<q.front()<<" ";

q.pop();

}

cout<<endl;

return 0;

}

10. Deletes the first element of the queue.

Sol – 10.

#include<bits/stdc++.h>

using namespace std;

int main()

{

queue<char>q;

q.push('u');

q.push('o');

q.push('g');

q.pop();

cout<<"Queue is"<<endl;

while(!q.empty())

{

cout<<q.front()<<" ";

q.pop();

}

cout<<endl;

return 0;

}