Data 1

Mariam Alaidy

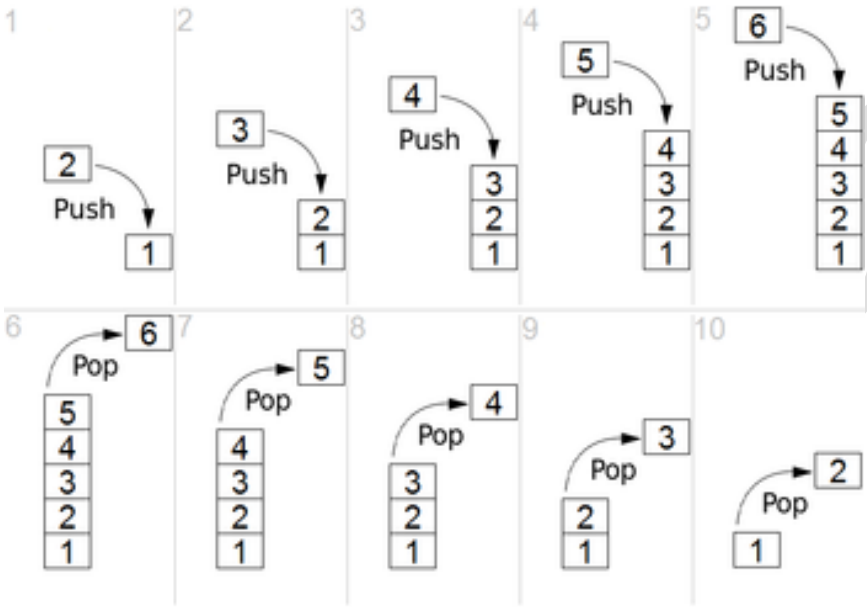
KASIT ACADIMIC TEAM
KASIT_TEAM
KASIT_TEAM
KASIT TEAM
KASIT TEAM
WWW.KASIT.NET
KASIT TEAM

The Standard Template Library (STL) by mariam alaidy

is a set of C++ template classes to provide common programming data structures and functions such as lists, stacks, arrays, etc. It is a library of container classes, algorithms, and iterators. It is a generalized library and so, its components are parameterized. A working knowledge of template classes is a prerequisite for working with STL.

STACK **(LIFO)**

(Last in Last out)



Stack is data structure that allows you to save values in a way that you can access only the last element you enter in it So if you want to access other element before last add you need to delete it

For example

The shape above show you how the memory will look like

Note * push mean you add the element to **stack** .

Pop you mean you delete(Last element you add it).

(1) push(1) to the stack [1]

then push(2) [1|**2**]

Note the only element you can access is 2 and if you want to access 1 you need to delete 2 [1|**2**]

(2) push(3) to stack [1|**2**] == [1|2|**3**]

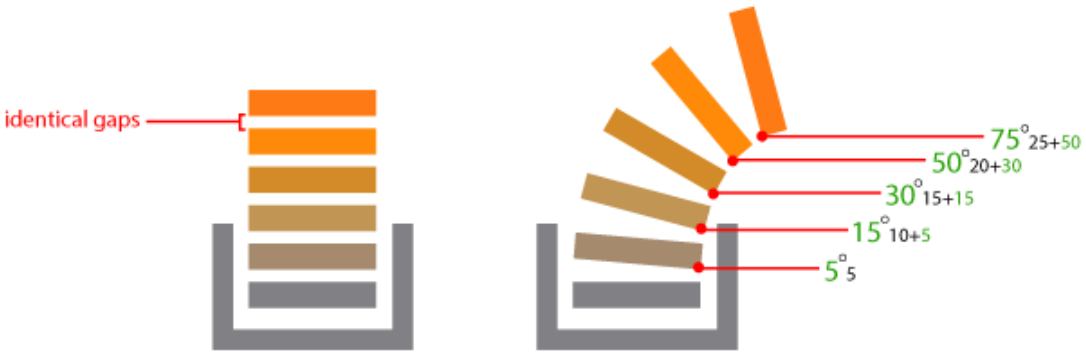
Note now you can only access 3 because it's the last element you added to stack

And if you want to access 1 you need to pop 3 == [1|**2**]

And pop 2 == [**1**]

(3) push(4) to stack [1|2|**3**]== [1|2|3|**4**]

And now only access able element is 4



This is another picture to make you understand how it works.

Stack STL code in c++

STL let you call bulid in stack in c++

And because it called standard **(template)** library

You can choose the data type you want to store values on it double , int ,string ,etc…

Stack have many functions that will helps you

push(); let you add element to stack

pop(); remove the last element you add

top(); return last element

size() return size of stack

empty() return Boolean true if empty false if there is element

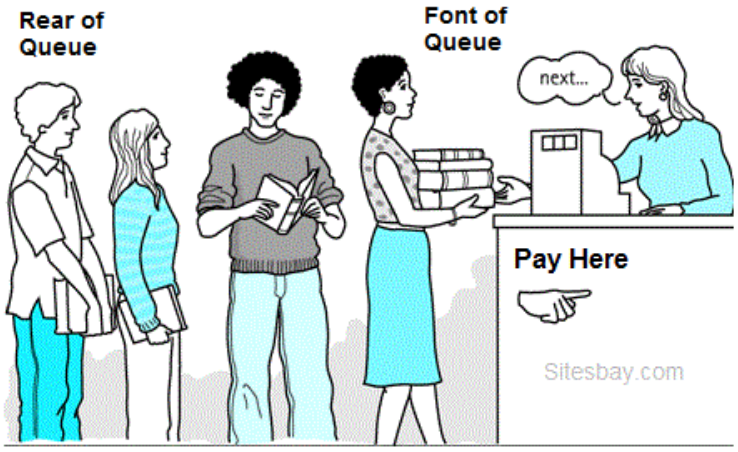
```cpp
#include<stack>//now you called the lib nome it's <stack> not "stack.h" not
the same you build before
#include<iostream>
using namespace std;
void print_stack(stack<int>e){//not with & called by value
	while(!e.empty()){
		cout<<e.top()<<" ";
		e.pop();
	}
}
int main(){
	stack<int>st;//<template> == data type
	/*
	stack<string>st1;
	stack<double>st2;
	stack<float>st3;
	etc...
	*/
	st.push(1); //[1]
	st.push(2); //[1|2]
	st.push(3);//[1|2|3]
	st.push(4);//[1|2|3|4]

	//there is no print function in build in stack so
	//there is 2 ways to print it
	//first one in the main but you will lose the data because of pop
	//every time

	//first way
	stack<int>stack2;
	stack2 = st;//because i will lose the elements i made another stack to
use it for second way

	while(!st.empty()){//check if it's not empty
		cout<<st.top()<<" ";//print the top
		st.pop();//pop the element
	}
	cout<<st.size();//0 because I delete the elements

	//second way write void function that take stack as pass by value
	//pass by value will print a copy of stack function(stack<int>e);
	//pass by refrence will print same stack function(stack<int>&r);
	//so we will choose pass by value in order to save the values

	//second way
	print_stack(stack2);
	cout<<stack2.size();//4 beacuse the function print_stack made a copy
from stack2 and print it without delete the main elements
	return 0 ;
}
```

Queue **(FIFO)**

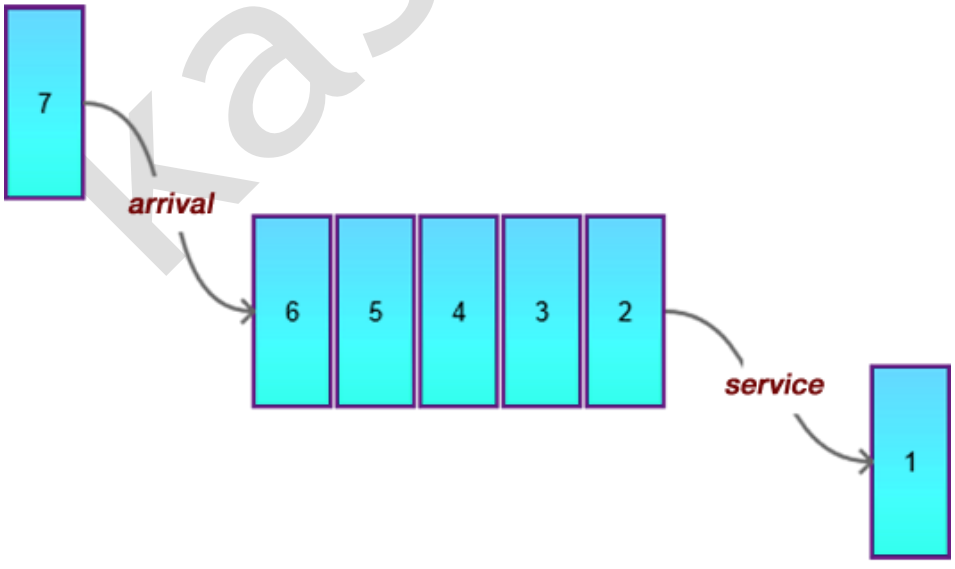
Queue is another data structure that allows you to add first element and still access it

People stand in queue for getting abook

First one who stand up to the queue get out from it



Real Life Example of Queue : Library Counter



Another example

This is empty queue []

Push(1) [1] the access element or front = 1 which is == **top() in stack**

push(2) → [2,**1] front =1**

push(3) -> [3,2,**1] front =1**

push(4) -> [4,3,2,**1] //**

push(5) ->[5,4,3,2**,1] //**

push(6) -> [6,5,4,3,2,**1] //**

pop() -> [6,5,4,3,**2]**

pop() -> [6,5,4,**3]**

pop() ->[6,5,**4]**

pop() -> [6,**5]**

pop() ->[**6**]

pop() -> []

Queue build in functions

Empty() return if true if queue is empty and false if it's not

Front() first element you add

Pop() delete first element you add to queue

//note

Function back() return last element you add but using it not efficient because pop

Delete first element not the last

For example

Queue[1,2,3,**4**] front = 4;back=1;

Pop Queue[1,2,**3**] front = 3 ;back=1;

Etc...

```
#include<queue>//calling the queue lib
#include<iostream>
using namespace std;
void print_queue(queue<int>q){//by value
	while(!q.empty()){
		cout<<q.front()<<" ";
		q.pop();
	}
}
int main(){
	queue<int>q;//create a queue
	q.push(1); //cout<<q.front(); == 1
	q.push(2);  //cout<<q.front();== 1
	q.push(3);
	q.push(4);
	//there is also for the same to stack
	//queue two ways to print
	//using normal queue pop in main and lose values
	//or create function print a
	//copy from it
	//pass by value
	print_queue(q);
	cout<<q.size();
//second way
	while(!q.empty()){
		cout<<q.front()<<" ";
		q.pop();
	}
	cout<<q.empty(); //1 means it's empty lost of data


}
```

Maps

Another data structure but it’s more different

For example when you open a dictionary

And searching for example one world (book)

You will **find only one meaning for it**

[**book**]= papers you can read from them

Another example every student have on ID

[**ID**] = student_name

In this case in c++ maps

ID , book called keys (because it can help you know the value for it)

Keys can be any **one of all the data types** and the values too

1120217	Nikhilesh
1120236	Navneet
1120250	Vikas
1120255	Doodrah

```cpp
#include<map>//calling the map lib
#include<iostream>
#include<string>
using namespace std;

int main(){
	//map<(key_data_type),(values_data_type)>(name_of_map);
	map<string  ,int>mp1;
	map<int,double>mp2;
	map<int,int>mp3;
	mp1["c++"]=1;
	mp1["java"]=2;
	//print the values not the keys
	cout<<mp1["c++"]; // it will print 1

	//

	mp2[333]=2.3;
	cout<<mp2[333]<<endl;
	cout<<mp2.size();

	//who to find it element has a value in the map
	if(mp1.find("m") ==mp1.end())
		cout<<"not";
	else cout<<"found";
	//how this is work ??
	//mp1 looks like( ["c++",1] , ["java",2])
	//in c++ there is something called pointer (Data2)
	//but to understand how it works
	//.end() point to the end of the map
	// mp1.end() points to ["java",2]
	//function find search for the element from first when it's not found
	//it means it get to the last element
	// ((# == search ))            # (["c++",1] , ["java",2])
	//start                        (# ["c++",1] , ["java",2]) notfound
	//                             (["c++",1] , #["java",2]) not found
	//.end() == ["java",2] and # == ["java",2] so it's not here beacuse it's reach
//the limit (.end()) so false


	//note it's not important to know this but this help you imagine it

	//you can't print maps without pointer (Data2)
	cout<<mp1.max_size()<<endl; //this is maximum space came from c++ comands (YOU
//JUST KNOW HOW TO PRINT IT)
mp1.erase("c++");//erase delete the value for key c++
//([c++,1])==([c++,EMPTY_])
	cout<<mp1["c++"]<<endl;


	mp1.clear();//delete every thing inside the mp1 for example  (["c++",1] ,
//["java",2],..)== ([],[],..)
}
```

Vector (dynamic arrays)

Why it’s calling like that when you define array

Int a[size];

You should write a size

But in vectors you just write and add element and remove element from it

And the size change dynamically

For example vector[] size =0;

[1,2,3] size =3;

[1] size =1;

Change according to the elements

```
#include<vector>
#include<iostream>
#include<string>
using namespace std;
void show_vector(vector<int >v){//print method2

	for(int i =0; i < v.size(); i++)
		cout<<v[i]<<" ";
	cout<<endl;

}
int main(){
	vector<int>vc;//define vector
	vc.push_back(1);
	vc.push_back(8);
	vc.push_back(77);
	cout<<vc.size()<<endl;//3
	vc.pop_back();
	cout<<vc.size()<<endl;//2

	for(int i = 0 ; i <vc.size();i++)
			cout<<vc[i]<<" ";

	//function resize
	vector <int> A;
	cout<<"size before "<<A.size()<<endl;
	A.resize(9,22);
	show_vector(A);
	cout<<"size after "<<A.size()<<endl;
	A.resize(14,3);
	show_vector(A);
	cout<<"size after "<<A.size()<<endl;
/*
	cout<<"front = "<<A.front()<< " back is "<<A.back()<<endl;
    cout<<"begin = "<<A.begin()<< " end is "<<A.end()<<endl;
	cout<<A.begin();
```

```
            cout<<"size of A is "<<A.size()<<endl;
            show_vector(A);
            CANT PRINT .being() because it'pointer
            */
            //cout<<"size of A is "<<A.size()<<endl;
            //show_vector(A);

      }
```

```
2D vector
[1] = [1,2,3,4]
[2] = [1,2,3]
[4] = [1]
Vector of vector ====== [[1]|[2]|[4]]
```

```
   #include<vector>
   #include<iostream>
   #include<string>
   using namespace std;
   void show_vector(vector<vector<int> >v){//print method2

        for(int i = 0 ; i < v.size() ;i++){
              //the size of all the vector [[1,2,3],[3,2]] size == 2
              //([1,2,3] == 1 , [3,2] == 1) so size 1+1 == 2



              //size of each vector of vector elements [[1|2|3] == size = 3 | [3,2] == size = 2]
              for(int j = 0 ; j <v[i].size() ;j++){
                    cout<<v[i][j]<<" ";//print [[0][1]] arrays
              }                          //rows [1][3]       the elements |
                                         //     [2][2]
              //                          >  colums([3])     <
              //
              // [1] == [0,0] ,[3] = [0,1] etc
              //[2] == [1,0] , [2] == [1,1]
              cout<<endl;
        }


   }
   int main(){
        vector<vector<int>>vc(2);//define vector of  vector == 2d arrays 2 == size
   //   vc.push_back(2);  error because you need to write which vector you want add to
        vc[0].push_back(2);//0 index of first [[2],..]
        vc[0].push_back(33);//[[2,33],..]
        vc[1].push_back(6);
        vc[1].push_back(444);//[[2,33],[6,444]]

        //to print it need function to print vector by vector
        vector<vector<int>>c(2);//2 size in case 2D write it
        c[0].push_back(1);
        c[0].push_back(2);
        c[0].push_back(3);
        c[1].push_back(1);
```

```
	c[1].push_back(1);
	show_vector(c);
}
```

Pair (X,Y)

The pair container is a simple consisting of two data elements or objects(FIRST,SECOND).

- The first element is referenced as ‘first’ and the second element as ‘second’ and the order is fixed (first, second).

How pair looks like in memory

[firstelement,secondelement]

For example

```
#include<iostream>
#include<string>
using namespace std;
int main(){
	pair<int,char>pr;//note we are calling pair class
	//you can choose any data type you want
	//for both options
	pr.first = 2;
	pr.second = 'a';
	cout<<pr.first<<" "<<pr.second<<'\n';//2 a
	//note this is one element imagine pair as datatype
	pair<char,double>pr2;
	pr2 = make_pair('a',33);//another way to full with values
	cout<<pr2.first<<' '<<pr2.second<<'\n';
	//
	//make many pair values we use pair as array
	pair<int,int>pr_array[4];
	//(pr_array[__,__],  [__,__],[__,__],[__,__])
	pr_array[0].first = 2;
	pr_array[0].second = 33;
	//pr_array[2,33],    [__,__],[__,__],[__,__])
	//another way
	//using loops
	for(int i = 0 ; i < 4 ; i++){
		pr_array[i]=make_pair(i,2);
	}
	for(int i = 0 ; i < 4 ; i++){
		cout<<pr_array[i].first<<" "<<pr_array[i].second<<'\n';
	}

}
```

#include<algorithms>Imagine you want to sort vector,maps,pair.

Its sometimes hard if it’ s have huge number of elements

So for that c++ create bulid in function to helps you do benefits stuff with such short time and easy implementation .

How to use them with arrays

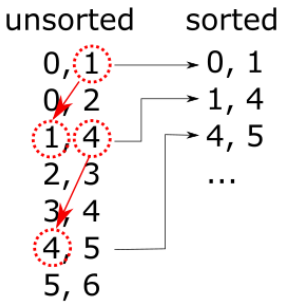
```
#include<iostream>
#include<algorithm>
using namespace std;
void print_array(int aa[]){
    for(int i = 0 ; i < 4 ; i++)
          cout<<aa[i]<<" ";
    cout<<endl;
}
int main(){
    int a[4] = {44,1,2,11};
    cout<<"print the array "<<endl;
    print_array(a);
    //sort(array name,array + size we want to sort)
    sort(a,a+2);//this will sort only first 2 elements
    //this will sort asceding way
    //smallest to largest
    //1 44 2 11
    cout<<"after sort only first two element "<<endl;
     print_array(a);
    //in case we want to sort all
    sort(a,a+4);
    //1 44 2 11
    cout<<"after sort all the array "<<endl;
     print_array(a);
     //sort dsceding way
     //largest to smallest
     //first sort it normal
     //the function reverse
     reverse(a,a+4);
     //this will reverse all the elements ->>  <--
     cout<<"after reverse"<<endl;
     print_array(a);
     //binary_search()
     //this is boolean function return true if you want to find element exits in the arry
     //false if not
     //example :
     int array_2[4]={1,2,4,5};
     //binary_search(name of array , name + size , element you want find );

     if(binary_search(array_2,array_2+4,7)){
           cout<<"found 7 in the array";
     }
     else cout<<"not found ";
     cout<<endl;
```

```
    if(binary_search(array_2,array_2+4,2)){
            cout<<"found 2 in the array";
    }
    else cout<<"not found ";
    cout<<endl;
}
```

How to use them with vector

```
#include<iostream>
#include<algorithm>
#include<vector>
using namespace std;
void print(vector<int>vc){
       for(int i = 0 ; i < vc.size() ;i++)
              cout<<vc[i]<<" ";
       cout<<endl;
}
int main(){
       vector<int>vc;
       vc.push_back(2);
       vc.push_back(4);
       vc.push_back(1);
       vc.push_back(0);
       //sort(name of vector with iterator begin to points in first elements,name of
vector with iterator end to points in end of the elements)
       cout<<"before sorting "<<endl;
       print(vc);
       cout<<"after sorting"<<endl;
       sort(vc.begin(),vc.end());
       print(vc);
       cout<<"after reversing ";
       reverse(vc.begin(),vc.end());
       print(vc);
}
```

How to use pair arrays




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

int main(){
       pair<int,int>pr[5];
       pr[0]=make_pair(1,2);
       pr[1]=make_pair(1,1);
       pr[2]=make_pair(0,9);
       pr[3]=make_pair(3,2);
       //sort array of pairs
       cout<<"before sorting "<<endl;
       for(int i = 0 ; i < 5 ; i++){
              cout<<pr[i].first<<" "<<pr[i].second<<'\n';
       }
       cout<<endl;
       sort(pr,pr+5);
```

```
	//how c++ sort pair
	//first compare first element
	//in case
	//[1,2],[1,1]
	//first element for both == 1
	//so it will choose it depends in the next element
	//[1,1] ,[1,2]
	cout<<"after sorting "<<endl;
	for(int i = 0 ; i < 5 ; i++){
		cout<<pr[i].first<<" "<<pr[i].second<<'\n';
	}

}
```

Exercises :

1//write a program the take stack consist of 5 numbers entered from the user and print the value only if number is even

```
#include<iostream>
#include<algorithm>
#include<stack>
using namespace std;
void print_even(stack<int>p){
	while(!p.empty()){
		if(p.top() % 2 == 0)
			cout<<p.top();
		p.pop();
	}
}
int main(){


	stack<int>st;
	int value;
	for(int i = 1 ; i < 5 ; i++){
		cin>>value;
		st.push(value);
	}

}
```

2//Do the same to queue

3//write program that push 3 9 100 23 in stack and create another stack and fill it with only odds number (its ok if first stack lost the values)

```
#include<iostream>
#include<algorithm>
#include<stack>
```

```
using namespace std;
void print(stack<int>p){
	while(!p.empty()){

			cout<<p.top()<<" ";
		p.pop();
	}
}

int main(){

	stack<int>st;
	st.push(3);
	st.push(9);
	st.push(100);
	st.push(23);

	stack<int>st2;
	while(!st.empty()){
		if(st.top() % 2 == 1)
			st2.push(st.top());
		st.pop();
	}
	print(st2);
}
```

4 write program that push(3,4,2,1,111,100) to queue
And fill it into stack
And print only evens

5-write pair arrays that first one is integer and second member is double and add
2 1.2
3 3.3
4 34.3
4 34.2
then sort it and print only the second values

```
#include<iostream>
#include<algorithm>
#include<stack>
using namespace std;
int main(){

	pair<int,double>pr[4];
	pr[0] = make_pair(2,1.2);
	pr[1]=make_pair(3,3.3);
	pr[2] = make_pair(4,34.3);
	pr[3] = make_pair(4,34.2);
	sort(pr,pr+4);
	for(int i = 0 ; i < 4 ; i++)
		cout<<pr[i].second<<" ";

}
```

NOTE these exercise seems hard a little so if you can’ t solve it it’ s ok

//write aprogram that will take queue of pairs and print only (seconds) if name == “sami”

//input

“sami” 3

“sami_” 4

“ahmad” 9

“laila” 3

```
#include<iostream>
#include<algorithm>
#include<queue>
#include<string>
using namespace std;
int main(){
	queue<pair<string,int >>q;//pair as datatype imagin it as array of class
	q.push(make_pair("sami",3));//make_pair for every pair class
	q.push(make_pair("sami",3));
	q.push(make_pair("ahmad",9));
	q.push(make_pair("laila",4));
	while(!q.empty()){
		if(q.front().first == "sami")//first is data type // front() function return
first element
			cout<<q.front().second<<" "<<endl;
		q.pop();
	}
}
```

//wirte aprogram that print (queue) in reverse way (3,2,10,3,22,100) //add value

NOTE function reverse can’t help in the stack or queue

//write a program that print even multiply by 10 in reverse way //(0,2,22,11,1,6)