String Operations

Length: str.size() or str.length()

Clear: str.clear()

Insert: str.insert(pos, new_str, len of new_str)

Substr: str.substr(position,len)

Erase: str.erase(position,len)

Comp: str1.compare(str2)

0=equal <0=first string less

Begin: str.begin(), str.end()

Repeat: str.insert(pos, no of times, char)

Replace: str.replace(pos,len,replacable_string)

Copy: str.copy(string_var, len, pos)

Find: str.find(search_string), + position, - std::string::npos

First: str.front()

Back: str.back()="new string"

Input: getline(cin, str_var)

Push: str.push_back(char)

Pop: str.pop_back() - deletes

Vector Operations

Add: vec.push_back(element)

Pop: vec.pop_back() - deletes

Erase: vec.erase(vec.begin()+pos)

Or vec.erase(vec.begin()+start_pos, vec.begin()+end_pos)

Clear: vec.clear()

Empty: vec.empty()

Iterator

Decl: vector<int> var(size,value)

Vector<int>::iterator it;

Value = *it

First: vector.begin()

Last: vector.end()

Insert: in between or at pos

Iter_var=vec_var.begin()+pos → begin is must

Vec_var.insert(iter_var, value)

Or Vec_var.insert(iter_var, no of times, value)

```
vector<int>::iterator temp=height.begin();
for(auto it=temp;it!=height.end(); it++)
{    cout<<*it; }
```

Type Casting

float x = (float) y + 2.3;

or

float x=

Unordered Set

Def: unordered_set<data_type> var1 = {"value1","value2"};

Insert: var1.insert(value1);

Erase: var1.erase(value1);

Size: var1.size();

Clear: var1.clear();

Find an element:

var1.count(value) // will return 1 if value there

else will return 0

2nd way

```
unordered_set<string>::iterator temp=var1.begin();
temp=var1.find(value);
if(temp==var1.end())
{  cout<<"not found":}
else
{ cout<<"found"<<*temp;}
```

Print:

```
for(auto it=var1.begin(); it!=var1.end(); it++)
{  cout<<*it;}
```

Unordered Map

Def: unordered_map<string,int>
temp={{"jeevan",25},{"babu",24}};

Insert: temp[“key1”]=value;

Erase: temp.erase(“key1”);

Size: temp.size();

Find:

temp.count(“Key”) //returns 1 if found

2nd way:

unordered_map<string,int>::iterator temp2=temp.begin();

temp2=temp.find("jeevan");

if(temp2==temp.end())

{ cout<<"not found";}

else

{ cout<<"found"<<temp2->first<<temp2->second;}

Print:

for(unordered_map<string,int>::iterator
it=temp.begin();it!=temp.end();it++)

{ cout<<it->first<<it->second<<endl;}

2nd way

for(auto& x: temp)

{cout<<x.first<<x.second<<endl;}

Stack

Def: stack<data_type> var;

Empty: var.empty(); // returns bool

Size: var.size()

Push: var.push(“value”);

Pop: var.pop() //remove last element

Top: var.top() //last element pushed

Swap: var.swap(var2) //exchange contents of var with var2

Queue

Def: queue<data_type> var;

Front: var.front(); //first element pushed

Last: var.back() //last element pushed