C++ And Linux

Fixed Startup Codes&Compiling&Executing

```
#include<iostream>
using namespace std;
int main(int argc,char *arg[])
{

}
```

g++ <file.cpp> <-o> <desired output name>
<./><output name><variables>

Pointers

<type><*><name>
<*><name>=value
<&><name>=adress=<name>

Pointer Arrays&Dynamic Memory Allocation:

<type><*><name>
<name>=<new><type><(size)>

Sample Code

```
#include<iostream>
using namespace std;
int main(int argc,char *arg[])
{
	int *p;
	int a=atoi(arg[1]);
	p=new int(a);
	for(int i=0;i<a;i++)
	{
		cout<<"Pointer Dinamik bellek ile dizi
"<<endl<<"*"<<*p<<endl<<"&"<<&p<<endl<<" "<<p<<endl;
		p++;
	}
	return 0;
}
```

Controlling And Ruling Errors (Try-Catch)

<try>
{
	Protected events
	<throw> “Throw message”;
}
<catch><(<const>< char*> <name>)>
{
	<cout><name>
}

Sample Code

```
#include<iostream>
using namespace std;
int main(int argc,char *arg[])
{
	int *p;
	int size=atoi(arg[1]);

	p=new int(size);

	cout<<"Causing some errors for pointers..."<<endl;

	cout<<"Causing overflow"<<endl;
	try
	{
		if(size>5)
		{
			throw "Size cannot be over 5";
		}
		try
		{
		for(int i=0;i<=(size+1);i++)
			p++;
			throw"Overflowed!";
		}
		catch(const char* msg)
		{
			cout<<endl<<"----------------Exception--------"<<endl;
			cerr<<msg<<endl;
		}

	}
	catch(const char* msg)
	{
		cout<<endl<<"----------------Exception--------"<<endl;
```

```
                cerr<<msg<<endl;
        }
}
```

Note: For throwing options its also possible that giving a variable type like integer char double etc. Which can be shown like:
<throw><value>;
catch(<type> <name>)
<cout><name>

Vector Usage

Vectors are same as dynamic arrays with ability to resize automatically

<vector><<type>><name>

<name>< . ><begin()> //Points the beginning of the vector
<name>< . ><end()> //Points the end of the vector
<name>< . ><push_back()> //Adds element to the end of the vector
<name>< . ><pop_back()> //Pops the last element of the vector
<name>< . ><size()> //Returns the size of the vector
<name>< . ><resize(<int>)> //Resizes the vector for more or less space
<name>< . ><at(<int>)> //Return the element value at desired location
<name>< . ><assign(<how many times><value>)> //Assigns an element replacing all others
<name>< . ><insert(<int position> , <value>)> //İnserts an element before the specified position
<name>< . ><erase(<int position>)> //Removes an item from specified position
<name>< . ><clear()> //Removes all items in the vector
<name>< . ><swap(<target vector>)> //Target vector must be the same type size can be different
<name>< . ><emplace(<position>,<value>)> //Adds new item to a specified location
<name>< . ><erase(<int position>)> //Erases specified item

Sample Code

```
#include<iostream>
#include<vector>
using namespace std;
void printer(vector<int>test,int size)
{
        for(auto i=test.begin();i!=test.end();i++)
                cout<<*i<<endl;
}
int main(int argc,char* arg[])
{
        vector<int> test;
        int size=atoi(arg[1]);
        for(int i=0;i<size;i++)
                test.push_back(i);
```

```
        cout<<endl<<"Max size : | "<<test.max_size()<<" | Current size: | "<<test.size()<<endl;

        test.resize((test.size()-1));

        cout<<endl<<"After resize without values size : |"<<test.size()<<" |"<<endl;

        printer(test,test.size());
        cout<<"Assigning 4 replacing old items"<<endl;
        test.assign(1,4);
        printer(test,test.size());
        cout<<"Assigning 5 replacing old items"<<endl;
        test.assign(1,5);
        printer(test,test.size());
        cout<<"size of the vector : "<<test.size()<<endl;
        cout<<"Inserting to beginning value :-1"<<endl;
        test.insert(test.begin(),-1);
        cout<<"Expected output: -1 5..."<<endl<<endl<<endl;
        printer(test,test.size());
        cout<<"clearing all items"<<endl;
        test.clear();
        cout<<"After clearing the size : "<<test.size()<<endl;

        printer(test,test.size());
        test.emplace(test.begin(),4);
        test.emplace(test.begin(),3);
        printer(test,test.size());
        cout<<"Erasing an item from marked location"<<endl;
        test.erase(test.end()-1);
        printer(test,test.size());

}
```