#include <iostream>

using namespace std;

int multiply(int a , int b);

int main()

{

cout<<multiply(4,5)<<endl;

cout<< [](int a, int b){return a\*b; }(4,5)<<endl;

auto f = [](int a , int b) {return a\*b; };

cout<<f(4,5)<<endl;

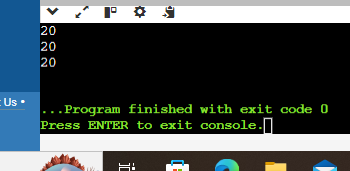
}

int multiply(int a , int b){

return a\*b;

}

Output:



#include <iostream>

using namespace std;

void lambda\_value\_capture() {

int value = 1;

auto copy\_value = [value] {

return value;

};

value = 100;

auto stored\_value = copy\_value();

cout<<"stored value "<< stored\_value <<endl;

}

int main()

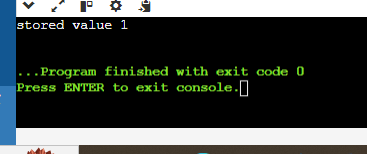
{

lambda\_value\_capture();

return 0;

}

Output:



#include <iostream>

using namespace std;

void lambda\_value\_capture() {

int value = 1;

auto copy\_value = [&value] {

return value;

};

value = 100;

auto stored\_value = copy\_value();

cout<<"stored value "<< stored\_value <<endl;

}

int main()

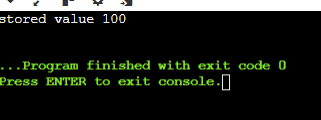
{

lambda\_value\_capture();

return 0;

}

Output:



#include <iostream>

using namespace std;

int main()

{

int m =0;

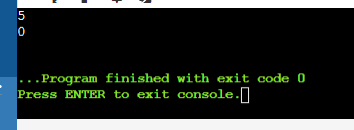
int n =0;

[&,n] (int a) mutable {m = ++n + a; }(4);

cout<< m <<endl << n <<endl;

}

Output:



#include <iostream>

#include <algorithm>

#include <vector>

using namespace std;

void assign(int& v)

{

static int n = 1;

v = n++;

}

void print(int v)

{

cout << v << " ";

}

int main()

{

vector<int> vec(10);

for\_each(vec.begin(), vec.end(), print);

cout<<endl;

for\_each(vec.begin(), vec.end(), assign);

for\_each(vec.begin(), vec.end(), print);

return 0;

}

Output:

