

Itish Agarwal  
18CS30021

Q1.

(a)  $[&]$  binds the free value by reference whereas  $[=]$  binds them by value.

$\therefore l(x)$  uses the updated value of  $x$  at time of invoking  $l(x)$ , whereas  $m(x)$  uses that value of  $x$  which is present at time of declaring  $m(x)$ .

$\therefore$  Output is:

35  
40  
12

~~(a) Output is: 6 8 8 11 11~~

~~(b) Output is: 6 8 8 11 11 (without spaces)~~

(b) output is: 6 8 8 11 11 (without spaces)

In  $f2$ , variables are accessed by reference, so changes made in  $f2$  is reflected outside  $f2$  as well. On the other hand,  $f1$  just maintains a local value of variable  $c$ .

(c) #include <iostream>  
using namespace std;  
int main() {

Itish Agarwal  
18CS30021

double F, C;

[&] () {

cout << "Enter temperature in fahrenheit:";

cin >> F;

$C = (F - 32.0) / (1.8);$

cout << "Temperature in Celcius is " << C << '\n';

} ();

return 0;

}

(d) #include <iostream>  
using namespace std;  
int main() {

int L, W, A, P;

[&] () {

cout << "Enter length (L) : ";

cin >> L;

cout << "Enter width (W): ";

cin >> W;

$A = L * W;$

$P = 2 * (L + W);$

cout << "Area of rectangle is : " << A << '\n';

cout << "Perimeter of rectangle is : " << P << '\n';

} ();

}

```
(2) #include <bits/stdc++.h>
using namespace std;
int main() {
```

Itish Agarwal  
18CS30021

```
cout << "Enter distance : ";  
cin >> dist;
```

```
int dist, cost = 0;
```

```
{
```

```
cout << "Enter distance : ";
```

```
cin >> dist;
```

```
cost = 100;
```

```
dist -= 12;
```

```
// For next 4 km
```

```
cost += max(0, min(dist, 4)) * 8;
```

```
dist -= 4;
```

```
// For next 4 km
```

```
cost += max(0, min(dist, 4)) * 6;
```

```
dist -= 4;
```

```
// For remaining distance
```

```
cost += max(0, dist) * 5;
```

```
// Print total cost
```

```
cout << "Total cost is : " << cost << "\n";
```

```
}();
```

```
return 0;
```

```
}
```

Q2.

(a) Code :

Itish Agarwal

18CS30021

```
#include <functional>
#include <bits/stdc++.h>
using namespace std;
```

```
class Get_Permutations {
public:
```

```
vector<string> operator () (string s) {
```

```
int n = s.size();
```

```
vector<string> ans, current;
```

```
string x = ""; // empty string
```

```
if (n == 0) {
```

```
current.push_back(x);
```

```
return current;
```

```
}
```

```
for (int i = 0; i < n; i++) {
```

```
current.clear();
```

```
x = "";
```

```
for (int j = 0; j < n; j++) {
```

```
if (i != j) {
```

```
x += s[j];
```

```
}
```

```
current = (*this)(x);
```

```
for (int j = 0; j < (int)current.size(); j++) {
```

```
ans.push_back(s[i] + current[j]);
```

```
}
```

```
}
```

```
return ans;
```



```
}  
}; //ending of GetPermutation class
```

```
int main () {  
    string s;  
    cin >> s;  
    int n = s.length();  
    GetPermutations gp;  
    vector<string> ans = gp(s);  
    cout << "Permutations of s are : '\n';  
    for (auto & it : ans) {  
        cout << it << '\n';  
    }  
    return 0;  
}
```

(b) PTO

(b) #include <iostream>  
#include <functional>  
using namespace std;

Itish Agarwal  
18CS30021

```
int main () {  
    string s;  
    cin >> s;  
    int n = s.size();  
    function<void(int, string)> rec = [l](int l, string s)  
    {  
        if (l == n-1) {  
            cout << s << '\n';  
            return;  
        }  
        for (int i = l; i < n; i++) {  
            swap(s[l], s[i]);  
            rec(l+1, s);  
            swap(s[l], s[i]);  
        }  
    };  
  
    rec  
    cout << "Permutations of s are: \n";  
    rec(0, s);  
    return 0;  
}
```