

Esercizio 1

```
1  int f(int x) {
2      return x * x + 4;
3  }
4  int main() {
5      int n;
6      cin >> n;
7
8      int p = 1;
9      for (int i = 1; i <= n; ++i) {
10         p = p * i;
11     }
12     cout << f(p) << end;
13     return 0;
14 }
```

Esercizio 2

```
1
2  void lte_5(int a[], int n) {
3      for (int i = 0; i < n; ++i) {
4          if (a[i] <= 5) {
5              return true;
6          }
7      }
8      return false;
9  }
10
11 void gt_5(int a[], int n) {
12     for (int i = 0; i < n; ++i) {
13         if (a[i] > 5) {
14             return true;
15         }
16     }
17     return false;
18 }
19
20 int main() {
21     const int N = 5;
22     int a[N];
23     for (int i = 0; i < N; ++i) cin >> a[i];
24
25     for (int i = 0; i < N; ++i) {
26         if (lte_5(a,N) or gt_5(a,N)) return 0;
```

```

27     }
28     return 0;
29 }

```

Esercizio 3

```

1  int main() {
2      int n;
3      cin >> n;
4
5      int *a = new int[n];
6
7      for (int i = 0; i < n; ++i) {
8          a[i] = 0;
9      }
10
11     int i = 0;
12     bool done = false;
13
14     while (not done) {
15
16         for (int k = 0; k < n; ++k) {
17             cout << a[i] << " ";
18         }
19         cout << endl;
20
21         if (i >= n) {
22             done = true;
23         } else {
24             if (a[i] < 4) {
25
26                 a[i]++;
27             } else {
28                 i++;
29             }
30         }
31     }
32
33     delete a[];
34     return 0;
35 }

```

Esercizio 4

```
1
2 bool somma(int M[][N]) {
3     for (int i = 0; i < N; ++i) {
4         if (M[i][i] != 0) {
5             for (int j = 0; j < N; ++j) {
6                 s += M[i][j];
7             }
8         }
9     }
10    return s > 0;
11 }
12
13 bool elabora(int M[][N], int V[N]) {
14     bool b = false;
15     int i = 1;
16
17
18     while (i <= N and not b) {
19         if (somma(M) || V[i-1] == 0) {
20             b = true;
21         }
22         i = i + 2;
23     }
24     return b;
25 }
```

Esercizio 5

```
1
2 bool g(int val, int &V[N]) {
3     bool b = false;
4     for (int i=0; i < N && !b; i++)
5         if (val == V[i])
6             b = true;
7     return b;
8 }
9
10 int f(int V1[N], int V2[N]) {
11     bool b = false;
12     int i = 1;
13     while (!b && i < N) {
14
15         b = g(V1[i], V2) && g(V2[i], V1);
```

```

16         i = i * 2;
17     }
18     return i;
19 }

```

Esercizio 6

#la funzione f1(M) ha complessità $O(N)$ e può terminare restituendo indifferentemente true o false

#la funzione f2(V) ha complessità $O(N^2)$ e può terminare restituendo indifferentemente true o false

```

1
2 void elabora(int M[][N], int V[N]) {
3     bool b = false;
4     int j, i = 0;
5     char car;
6
7
8     while (i<N && !b) {
9         cin >> car;
10        switch (car) {
11            case 'a':
12
13                if (f2(V))
14                    for (j=0; j<N; j++)
15                        cout << M[i][j];
16                break;
17            case 'b':
18
19                if (f1(M) && f2(V))
20                    b = true;
21                else
22                    cout << V[i];
23                break;
24            default:
25                cout << V[i];
26        }
27
28        for (int x=0; x<N; x++)
29            if (M[x][i] != V[x])
30                b = true;
31        i++;
32    }
33 }

```