```cpp
//Algoritmo de busca binaria
int busca_binaria( int vet[], int chave, int tam){
  int inicio = 0;  //inicia o indice do vetor na pos 0
  int fim = tam-1; //ultima posição do vetor
  int meio;
  while (inicio <= fim){
    meio = (inicio+fim)/2;
    if (chave == vet[meio]) return meio;
    if (chave < vet[meio]) fim = meio-1;
    else inicio = meio+1;
  }
  return -1; //o elemento procurado não esta no vetor
}

/*
  Declarar:
  int PI[MAX], pi;
*/
void crivo(int n) {
  int i, j, primo, raiz;
  PI[0] = 2; pi = 1;
  for (i = 3; i <= n; i++) {
    primo = 1; raiz = sqrt(i);
    for (j = 0; primo && j < pi && PI[j] <= raiz; j++)
      if (i % PI[j] == 0) primo = 0;
    if (primo) PI[pi++] = i;
  }
}

//MDC
int gcd(int a, int b) { return (b == 0) ? a : mdc(b, a % b); }
int lcm(int a, int b) { return a * (b / gcd(a, b)); }

//EXPONENCIAÇÃO BINARIA
long long expbin(long long a, long long b, long long m) {
  long long y;
  if (b == 0) return 1;
  if (b & 1) return a * expbin(a, b - 1, m) % m;
  y = expbin(a, b >> 1, m);
  return y * y % m;
}

ll sum_div(int n) {
  ll id_pf = 0, pf = PI[0], resp = 1, pot; //pot é potencia
  while (pf * pf <= n) {
    pot = 0;
    while (n % pf == 0) { n /= pf; pot++; }
    resp *= (expbin(pf, pot + 1) - 1) / (pf - 1);
    pf = PI[++id_pf];
  }
  if (n != 1) resp *= (expbin(n, 2) - 1) / (n - 1);
  return resp;
}

ll numDiv(ll n) {
  ll id_pf = 0, pf = PI[0], resp = 1, pot; //pot é potencia
  while (pf * pf <= n) {
    pot = 0;
    while (n % pf == 0) { n /= pf; pot++; }
    resp *= (pot + 1);
    pf = PI[++id_pf];
  }
  if (n != 1) resp *= 2;
  return resp;
}


/*  Declarar struct para fatorar, e um id para cada fator primo */
void fatora(ll n) {
  int tmp;
  ll id_pf = 0, pf = PI[0];
  id_fator = 0;
  while (pf * pf <= n) {
    tmp = 0;
    while (n % pf == 0) { n /= pf;  tmp++; }
    if (tmp) {
      fatores[id_fator].fp = PI[id_pf];
      fatores[id_fator++].m = tmp;
    }
```

```
80        pf = PI[++id_pf];
81      }
82      if (n != 1) {
83        fatores[id_fator].fp = n;
84        fatores[id_fator++].m = 1;
85      }
86    }
```