TP 1 - Partie 2 - Exercice 8 - I

```
#include <cassert>
#include <cmath>
#include <iostream>

#define SHOW(arg) std::cout << "Macro SHOW “" #arg "”: " << (arg) << '\n';

// Returns poly[0] + poly[1]*x^1 + ... + poly[n-1]*x^(n-1)
double horner_iterative_eval(double const poly[], int n, double x) {
  assert(n > 0);
  --n;
  double result{poly[n]};
  while (--n >= 0)
    result = result * x + poly[n];
  return result;
}
double norminv(double x) {
  assert((0 <= x) && (x <= 1));
  double constexpr a[]{2.50662823884, -18.61500062529, 41.39119773534,
                       -25.44106049637};
```

TP 1 - Partie 2 - Exercice 8 - II

```cpp
    int constexpr n_a{sizeof a / sizeof a[0]};
    // b_0 is equal to 1.
    double constexpr b[]{1, -8.47351093090, 23.08336743743, -21.06224101826,
                         3.13082909833};
    int constexpr n_b{sizeof b / sizeof b[0]};
    double constexpr c[]{
        0.3374754822726147, 0.9761690190917186, 0.1607979714918209,
        0.0276438810333863, 0.0038405729373609, 0.0003951896511919,
        0.0000321767881768, 0.0000002888167364, 0.0000003960315187};
    int constexpr n_c{sizeof c / sizeof c[0]};
    double const y{x - .5};
    // Central region.
    if (std::abs(y) < .42)
      return y * horner_iterative_eval(a, n_a, y * y) /
             horner_iterative_eval(b, n_b, y * y);
    // The tails.
    double const r{(y < 0) ? x : 1 - x};
    double const s{std::log(-std::log(r))};
    double const t{horner_iterative_eval(c, n_c, s)};
    return (x > .5) ? t : -t;
```

TP 1 - Partie 2 - Exercice 8 - III

```cpp
}
int main() {
  std::cout << "-inf    expected - ", SHOW(norminv(0));
  std::cout << "-1.6449 expected - ", SHOW(norminv(.05));
  std::cout << "-1.2816 expected - ", SHOW(norminv(.1));
  std::cout << " 0.0000 expected - ", SHOW(norminv(.5));
  std::cout << " 1.2816 expected - ", SHOW(norminv(.9));
  std::cout << " 1.6449 expected - ", SHOW(norminv(.95));
  std::cout << "+inf    expected - ", SHOW(norminv(1));
}
```

Output:

```
-inf    expected - Macro SHOW “norminv(0)”: -inf
-1.6449 expected - Macro SHOW “norminv(.05)”: -1.64485
-1.2816 expected - Macro SHOW “norminv(.1)”: -1.28155
 0.0000 expected - Macro SHOW “norminv(.5)”: 0
 1.2816 expected - Macro SHOW “norminv(.9)”: 1.28155
 1.6449 expected - Macro SHOW “norminv(.95)”: 1.64485
+inf    expected - Macro SHOW “norminv(1)”: inf
```