The experienced C programmer will probably be a bit frustrated with the attention to details required by the Ada compiler. You will not have your favorite "tricks" available to fool the compiler into doing something out of the ordinary. The Ada compiler cannot be fooled. - http://perso.telecom-paristech.fr/~pautet/Ada95/intro.htm

For this programming assignment, you will be writing an ada program that primarily works with binary arrays. A binary array is simply an array that is of length 16 that can ONLY hold 1's and 0's. You will need to implement the following procedures and functions (recall that procedures ONLY work with parameters and don't return anything, while functions return data), whose prototypes have been provided to you in the assgn.ads package (this file is kind of like a header file in c, you will need to define these protocols in assgn.adb).

1) Procedure Init_Array. Accepts a two-way parameter (in-out mode) that is a BINARY_ARRAY and fills it with an initial set of random values. I have provided a second package to you, as a convenience, that you can use to generate random values with. Keep in mind as you generate this binary array that generating a binary value too large could result in an overflow when attempting to add two binary arrays together.
2) Procedure Print_Bin_Array. This procedure accepts a BINARY_ARRAY (in-mode only) and simply prints it to the console.
3) Procedure Reverse_Bin_Arr. This procedure accepts a BINARY_ARRAY (in-out mode) parameter and simply reverses that BINARY_ARRAY.
4) Function Int_To_Bin. This function accepts an INTEGER value (in-mode only) and returns the equivalent BINARY_ARRAY
5) Function Bin_To_Int. This function accepts a BINARY_ARRAY value (in-mode only) and returns the equivalent INTEGER.

Overloaded functions: When doing addition or subtraction for the overloaded operators below, you MUST perform BINARY addition or subtraction (adding or subtracting 1's and 0's one position at a time, and carrying, if necessary. Do NOT simply convert to integer values and add or subtract integers.

6) Overloaded "+" operator that accepts an INTEGER type and a BINARY_ARRAY type (in-mode only), adds them together and returns a BINARY_ARRAY that is the result.

7) A second Overloaded "+" operator that accepts two BINARY_ARRAY types (in-mode only), adds them together and returns the result as a BINARY_ARRAY
8) Two Overloaded "–" operators that behave like #6 and #7 above.

I've provided you with a main.adb file (an ada file with a main method in it) that you can use to test your code.

Helpful Links:

For linux systems you can simply install gnat-X, where X is some version number, to get all of the tools you need to compile and link ada programs.

Download here. You want GNAT ada 2016: http://libre.adacore.com/download/

Old but very useful (careful with the chapter on packages. Newer versions of ada require you to write them in two separate files. The specification belongs in .ads, and the implementation belongs in .adb):

http://perso.telecom-paristech.fr/~pautet/Ada95/intro.htm

Other Useful Links:

http://www.cs.fsu.edu/~baker/ada/examples/

http://www.cristhianny.com/others/ada_tutorial_introduction_code.html

http://www.adapower.com/adapower1/articles/class.html