Steps taken to design the program:

1. Focus on the easy functions first that I know would take very little effort and build towards the end goal (leap year, days in month, checking validity, prompting for calendar info…)
2. After these functions were complete I worked on the most essential function the build calendar function the plan i had in my head for this function was using the hint of 4x3 structure where I would do the first 3 months then have a new row for next months so I developed a grid that would store the correct information for the months aka the weeks and how many columns we need in store for all the dates. As weeks make our rows and columns are just our columns so I needed a 2d array with (1-6 weeks and 7 days x 3 months for columns). So in that function I knew that when I needed a nested for loop so the first one would iterate through the months and fill the correct info second one through the columns.
3. After that essential step the rest would flow nicely since they aren't very difficult. Except for printing the header of the calendar is where I would spend the most time on this program. First thing I thought was to make the year a string and to get the height and the width of the numbers that we are making the assignment say 10 high and then I did 7 wide. Then I made the file with all the number fonts and then stored them in a 3d array with a certain number including the height and width. This loop would work by storing for example 0, then looking at lines 1 through 10 and the width of each line until all 10 lines are finished then we iterate the first loop to 1 (height and width) consistency was everything. then went back to the function. Here I would iterate through the our digits in our string which is the year then convert it back to an integer and then use a loop to print out our fontarray that takes in the values of the digit we want to use the i'th line of the loop and the column, the essential part here was understanding to print it row by row for example:

   1 77777 999 1
   1 7 9 9 1 etc…
4. After this we just build our main with a simple loop that goes through the months and day and just using our functions

Yes, Ada was highly effective for this project. Because a calendar is essentially a grid, it requires a language that prioritizes precision and data integrity. Ada is a math based language which makes it a lot easier to sort through issues at runtime and compilation are quickly caught, so logic errors were much less common when designing this program. This made the development process much more stable compared to a more flexible language like C or Python, where a small indexing mistake might just result in messy output. The big benefit of using ada for this assignment was the declaration of the arrays as this made it very simple to store the font and the barebones grid for the calendar and header. The in out feature for this structure was vital for the dayOffset. By passing it as in out to the buildcalendar procedure, Ada handled the starting month day across the entire year perfectly making our indexing very neat. Ada syntax is very readable, helping me know what I am doing at all points and what I am missing. For example loop end loop and else end if statements make it very apparent what is trying to be done. While this means more typing, it makes the logic of complex nested loops very easy to read. As for limitations ada felt very nice to code in and did not feel like i was running into language specific problems as far as I was concerned. Everything was done in a neat fashion and things that I thought were hard come from a lack of understanding what needed to be done rather than the language itself.