Write a program that prints ‘Hello World’ to the screen.

Write Program to Find ASCII Value of a Character

Write [Program to Compute Quotient and Remainder](https://www.programiz.com/c-programming/examples/remainder-quotient)

Write Program to Swap Two Numbers using third variable

Write Program to Swap Two Numbers without using third variable

Write Program to Check Whether a Number is Even or Odd, same for finding between a range

Write Program to Find the Largest Number Among Three Numbers

Write Program to Check Leap Year

Write Program to Calculate the Sum of Natural Numbers

Write Program to Find Factorial of a Number

Write Program to Display Fibonacci Sequence

Write Program to Display Fibonacci Sequence in reverse order if last two numbers are given

write Program to Find GCD of two Numbers

write Program to Find LCM of two Numbers

[Program to Display Characters from A to Z Using Loop](https://www.programiz.com/c-programming/examples/display-alphabets)

[Write Program to Count Number of Digits in an Integer](https://www.programiz.com/c-programming/examples/digits-count)

[Write Program to Reverse a Number](https://www.programiz.com/c-programming/examples/reverse-number)

[Write a Program to Calculate the Power of a Number](https://www.programiz.com/c-programming/examples/power-number)

Write Program to Calculate the Power of a Number without using \* operator, you can use the plus operator

Write Program to Check Whether a Number is Palindrome or Not

| Write Program to Check Whether a Number is Prime or Not |
| --- |
| Write Program to Display Prime Numbers Between Two Intervals |
| Write Program to Check Armstrong Number |
| Write Program to Display Armstrong Number Between Two Intervals |
| Write Program to Display Factors of a Number |
| Write Programming Code To Create a Pyramid and Structure |
| Write Program to Make a Simple Calculator Using switch...case |
| Write Program to Display Prime Numbers Between Intervals Using Function |
| Write Program to Check Prime or Armstrong Number Using User-defined Function |
| Write Program to Check Whether a Number can be Expressed as Sum of Two Prime Numbers |
| Write Program to Find the Sum of Natural Numbers using Recursion |
| Write Program to Find Factorial of a Number Using Recursion |
| Write Program to Find G.C.D Using Recursion |
| Write Program to Convert Binary Number to Decimal and vice-versa |
| Write Program to Convert Octal Number to Decimal and vice-versa |
| Write Program to Convert Binary Number to Octal and vice-versa |
| Write Program to Reverse a Sentence Using Recursion |
| Write Program to calculate the power using recursion |
| Write Program to Calculate Average Using Arrays |
| Write Program to Find the Largest and Smallest Element of an Array |
| Write Program to Calculate Standard Deviation |
| Write Program to Addition Two Matrix Using Multi-dimensional Arrays |
| Write Program to Multiply to Matrix Using Multi-dimensional Arrays |
| Write Program to Find Transpose of a Matrix |
| Write Program to Multiply two Matrices by Passing Matrix to a Function |
| Write Program to Access Elements of an Array Using Pointer |
| Write Program Swap Numbers in Cyclic Order Using Call by Reference |
| Write a Program to Find the Largest Number Using Dynamic Memory Allocation |
| Write Program to Find the Frequency of Characters in a String |
| Write Program to Count the Number of Vowels, Consonants and so on |
| Write Program to Remove all Characters in a String Except Alphabet |
| Write Program to Find the Length of a String |
| Write Program to Concatenate Two Strings |
| Write Program to Copy String Without Using strcpy() |
| Write Program to Sort Elements in Lexicographical Order (Dictionary Order) |
| Write Program to Store Information of a Student Using Structure |
| Write Program to Add Two Distances (in inch-feet) System Using Structures |
| Write Program to Add Two Complex Numbers by Passing Structure to a Function |
| Write Program to Calculate Difference Between Two Time Periods |
| Write Program to Store Information of Students Using Structure |
| Write Program to Store Information Using Structures with Dynamically Memory Allocation |
| Write Program to Write a Sentence to a File |
| Write Program to Read a Line From a File and Display it |
| Write Program to Display its own Source Code as Output |
| Write Programming Code To Create a Pyramid and Pattern |

Write a program that asks the user for their name and greets them with their name.

Modify the previous program such that only the users Alice and Bob are greeted with their names.

Write a program that asks the user for a number n and prints the sum of the numbers 1 to n

Modify the previous program such that only multiples of three or five are considered in the sum, e.g. 3, 5, 6, 9, 10, 12, 15 for n=17

Write a program that asks the user for a number n and gives them the possibility to choose between computing the sum and computing the product of 1,…,n.

Write a program that prints a multiplication table for numbers up to 12.

Write a program that prints all prime numbers. (Note: if your programming language does not support arbitrary size numbers, printing all primes up to the largest number you can easily represent is fine too.)

Write a guessing game where the user has to guess a secret number. After every guess, the program tells the user whether their number was too large or too small. At the end, the number of tries needed should be printed. It counts only as one try if they input the same number multiple times consecutively.

Write a program that prints the next 20 leap years.

Middle Level

Write a function that reverses a list, preferably in place.

Write a function that checks whether an element occurs in a list.

Write a function that returns the elements on odd positions in a list.

Write a function that computes the running total of a list.

Write a function that tests whether a string is a palindrome.

Write three functions that compute the sum of the numbers in a list: using a for-loop, a while-loop and recursion. (Subject to availability of these constructs in your language of choice.)

Write a function on\_all that applies a function to every element of a list. Use it to print the first twenty perfect squares. The perfect squares can be found by multiplying each natural number with itself. The first few perfect squares are 1\*1= 1, 2\*2=4, 3\*3=9, 4\*4=16. Twelve, for example, is not a perfect square because there is no natural number m so that m\*m=12. (This question is tricky if your programming

language makes it difficult to pass functions as arguments.)

Write a function that concatenates two lists. [a,b,c], [1,2,3] → [a,b,c,1,2,3]

Write a function that combines two lists by alternatingly taking elements, e.g. [a,b,c], [1,2,3] → [a,1,b,2,c,3].

Write a function that merges two sorted lists into a new sorted list. [1,4,6],[2,3,5] → [1,2,3,4,5,6]. You can do this quicker than concatenating them followed by a sort.

Write a function that rotates a list by k elements. For example [1,2,3,4,5,6] rotated by two becomes [3,4,5,6,1,2]. Try solving this without creating a copy of the list. How many swap or move operations do you need?

Write a function that computes the list of the first 100 Fibonacci numbers. The first two Fibonacci numbers are 1 and 1. The n+1-st Fibonacci number can be computed by adding the n-th and the n-1-th Fibonacci number. The first few are therefore 1, 1, 1+1=2, 1+2=3, 2+3=5, 3+5=8.

Write a function that takes a number and returns a list of its digits. So for 2342 it should return [2,3,4,2].

Write functions that add, subtract, and multiply two numbers in their digit-list representation (and return a new digit list). If you’re ambitious you can implement Karatsuba multiplication. Try different bases. What is the best base if you care about speed? If you couldn’t completely solve the prime number exercise above due to the lack of large numbers in your language, you can now use your own library for this task.

Write a function that takes a list of numbers, a starting base b1 and a target base b2 and interprets the list as a number in base b1 and converts it into a number in base b2 (in the form of a list-of-digits). So for example [2,1,0] in base 3 gets converted to base 10 as [2,1].

Implement the following sorting algorithms: Selection sort, Insertion sort, Merge sort, Quicksort, Stooge Sort. Check Wikipedia for descriptions.

Implement binary search.

Write a function that translates a text to Pig Latin and back. English is translated to Pig Latin by taking the first letter of every word, moving it to the end of the word and adding ‘ay’. “The quick brown fox” becomes “Hetay uickqay rownbay oxfay”.

Write a function that returns the largest element in a list.

**Intermediate Level**

Write a program that takes the duration of a year (in fractional days) for an imaginary planet as input and produces a leap-year rule that minimizes the difference to the planet’s solar year.

Implement a data structure for graphs that allows modification (insertion, deletion). It should be possible to store values at edges and nodes. It might be easiest to use a dictionary of (node, edge list) to do this.

Write a function that generates a DOT representation of a graph.

Write a program that automatically generates essays for you.

Using a sample text, create a directed (multi-)graph where the words of a text are nodes and there is a directed edge between u and v if u is followed by v in your sample text. Multiple occurrences lead to multiple edges.

Do a random walk on this graph: Starting from an arbitrary node choose a random successor. If no successor exists, choose another random node.

Write a program that automatically converts English text to Morse code and vice versa.

Write a program that finds the longest palindromic substring of a given string. Try to be as efficient as possible!

Think of a good interface for a list. What operations do you typically need? You might want to investigate the list interface in your language and in some other popular languages for inspiration.

Implement your list interface using a fixed chunk of memory, say an array of size 100. If the user wants to add more stuff to your list than fits in your memory you should produce some kind of error, for example, you can throw an exception if your language supports that.

Improve your previous implementation such that an arbitrary number of elements can be stored in your list. You can, for example, allocate bigger and bigger chunks of memory as your list grows, copy the old elements over and release the old storage. You should probably also release this memory eventually if your list shrinks enough not to need it anymore. Think about how much bigger the new chunk of memory should be so that your performance won’t be killed by allocations. Increasing the size by 1 element, for example, is a bad idea.

If you chose your growth right in the previous problem, you typically won’t allocate very often. However, adding to a big list sometimes consumes considerable time. That might be problematic in some applications. Instead, try allocating new chunks of memory for new items. So when your list is full and the user wants to add something, allocate a new chunk of 100 elements instead of copying all elements over to a new large chunk. Think about where to do the book-keeping about which chunks you have. Different bookkeeping strategies can quite dramatically change the performance characteristics of your list.

Implement a binary heap. Once using a list as the base data structure and once by implementing a pointer-linked binary tree. Use it for implementing heap-sort.

Implement an unbalanced binary search tree.

Implement a balanced binary search tree of your choice. I like (a,b)-trees best.

Compare the performance of insertion, deletion and search on your unbalanced search tree with your balanced search tree and a sorted list. Think about good input sequences. If you implemented an (a,b)-tree, think about good values of a and b.

**Advance Level**

Given an array with numbers, write a program that efficiently answers queries of the form: “Which is the nearest larger value for the number at position i?”, where distance is the difference in array indices. For example in the array [1,4,3,2,5,7], the nearest larger value for 4 is 5. After linear time preprocessing you should be able to answer queries in constant time.

Given two strings, write a program that outputs the shortest sequence of character insertions and deletions that turn one string into the other.

Write a function that multiplies two matrices together. Make it as efficient as you can and compare the performance to a polished linear algebra library for your language. You might want to read about Strassen’s algorithm and the effects CPU caches have. Try out different matrix layouts and see what happens.

Implement a van Emde Boas tree. Compare it with your previous search tree implementations.