Prime:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **F000** | 21, 00, F1 |  | LXI H,F100 | Point to F100 to take the number |
| **F003** | FE |  | MOV A,M | Take the number into Accumulator |
| **F004** | 0E |  | MVI C,00 | Clear C register |
| **F006** | 57 |  | MOV D,A | Copy A to D |
| **F007** | 5F |  | MOV E,A | Copy A to E |
| **F008** | 42 | L2 | MOV B,D | Load B with D |
| **F009** | B8 | L1 | CMP B | Compare B with A |
| **F00A** | DA, 11, F0 |  | JC LABEL | if carry is generated, jump to Label |
| **F00D** | 90 |  | SUB B | Subtract B from A |
| **F00E** | C2, 09, F0 |  | JNZ L1 | Jump to L1 |
| **F011** | FE, 00 | LABEL | CPI 00 | Compare A with 00H |
| **F013** | C2, 17, F0 |  | JNZ SKIP | If Z = 0, jump to SKIP |
| **F016** | 0C |  | INR C | Increase C by 1 |
| **F017** | 7B | SKIP | MOV A,E | Load A with E again |
| **F018** | 15 |  | DCR D | Decrease D by 1 |
| **F019** | C2, 08, F0 |  | JNZ L2 | Jump to L2 label if Z = 0 |
| **F01C** | 79 |  | MOV A,C | Load C to A |
| **F01D** | 32, 01, F1 |  | STA F101 | Store result into F101 |
| **F020** | 76 |  | HLT | Terminate the program |

Binary search

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **F000** | 3A, 00, F1 |  | LDA F100 | Load the key into A |
| **F003** | 47 |  | MOV B,A | Store key into B |
| **F004** | AF |  | XRA A | Clear Acc |
| **F005** | 32, 03, F1 |  | STA F103 | Store number of iterations into F103H |
| **F008** | 6F |  | MOV L,A | Store A into L also |
| **F009** | 26, 09 |  | MVI H,09 | Load 9 into H |
| **F00B** | 3A, 03, F1 | START | LDA F103 | load number of iterations into A |
| **F00E** | 3C |  | INR A | Increase A |
| **F00F** | 32, 03, F1 |  | STA F103 | Restore number of iterations |
| **F012** | 7C |  | MOV A,H | Take the upper limit from H to A |
| **F013** | BD |  | CMP L | Compare A and L |
| **F014** | DA, 46, F0 |  | JC L2 | If CY = 1, jump to L2 |
| **F017** | 85 |  | ADD L | Otherwise add L with A |
| **F018** | 1F |  | RAR | Right rotate to get half of it |
| **F019** | 4F |  | MOV C,A | Store mid value into C |
| **F01A** | D2, 1E, F0 |  | JNC RESET | If CY = 0, jump to Reset |
| **F01D** | 3F |  | CMC | Complement the carry |
| **F01E** | 11, 10, F1 | RESET | LXI D,F110 | Load the initial address of array into DE |
| **F021** | 83 |  | ADD E | Add E and Mid |
| **F022** | 5F |  | MOV E,A | Store index into E |
| **F023** | AF |  | XRA A | Clear A |
| **F024** | 8A |  | ADC D | Add D with A with the carry |
| **F025** | 57 |  | MOV D,A | Restore A into D |
| **F026** | 1A |  | LDAX D | Load A with the value of mid position |
| **F027** | B8 |  | CMP B | compare it with Key |
| **F028** | DA, 34, F0 |  | JC ELSE | If CY = 1, jump to ELSE |
| **F02B** | CA, 3A, F0 |  | JZ PRINT | If they are same, jump to PRINT |
| **F02E** | 79 |  | MOV A,C | Take the mid from C to A |
| **F02F** | 3D |  | DCR A | Decrease A to get mid – 1 |
| **F030** | 67 |  | MOV H,A | Update the upper limit with mid – 1 |
| **F031** | C3, 0B, F0 |  | JMP START | Jump to START again |
| **F034** | 79 | ELSE | MOV A,C | Load the mid from C to A |
| **F035** | 3C |  | INR A | increase A to get mid + 1 |
| **F036** | 7D |  | MOV A,L | update lower limit with mid + 1 |
| **F037** | C3, 0B, F0 |  | JMP START | Jump to START again |
| **F03A** | 3E, 01 | PRINT | MVI A,01 | Load 1 as item is found |
| **F03C** | 32, 01, F1 |  | STA F101 | Store result at F101 |
| **F03F** | 79 |  | MOV A,C | Take the mid from C to A |
| **F040** | 32, 02, F1 |  | STA F102 | Store index of the key into F102 |
| **F043** | C3, 4B, F0 |  | JMP END | End the task |
| **F046** | 3E, 02 | L2 | MVI A,02 | Load 2 into A that key not present |
| **F048** | 32, 01, F1 |  | STA F101 | Store result at F102 |
| **F04B** | 76 | END | HLT | Terminate the program |

Selelction sort

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **F000** | 31, 00, 90 |  | LXI SP,9000H | Initialize Stack Pointer |
| **F003** | 21, 00, 80 |  | LXI H,8000H | Point to get the block size |
| **F006** | 4E |  | MOV C,M | Get the count |
| **F007** | 23 | DO | INX H | Point to next location to get block |
| **F008** | 7E |  | MOV A,M | Load the element to A |
| **F009** | CD, 18, F0 |  | CALL MIN | Find the minimum |
| **F00C** | BE |  | CMP M | Compare M and A |
| **F00D** | CA, 13, F0 |  | JZ GO | if Z = 1, they are same, skip swapping |
| **F010** | CD, 2A, F0 |  | CALL SWAP | Swap minimum and current content |
| **F013** | 0D | GO | DCR C | Decrease C by 1 |
| **F014** | C2, 07, F0 |  | JNZ DO | If Z = 0, go to Do |
| **F017** | 76 |  | HLT | Terminate the program |
| **F018** | E5 | MIN | PUSH H | Push HL into Stack |
| **F019** | C5 |  | PUSH B | Push BC into stack |
| **F01A** | 0D |  | DCR C | Decrease C by 1 |
| **F01B** | 23 | LOOP | INX H | Point to next location |
| **F01C** | BE |  | CMP M | Compare memory data with A |
| **F01D** | DA, 23, F0 |  | JNC SKIP | If CY = 0, jump to SKIP |
| **F020** | 7E |  | MOV A,M | Update the value of A |
| **F021** | 54 |  | MOV D,H | Copy H to D |
| **F022** | 5D |  | MOV E,L | Copy L to E |
| **F023** | 0D | SKIP | DCR C | Decrease C by 1 |
| **F024** | C2, 1B, F0 |  | JNZ LOOP | If Z = 0, go to Loop |
| **F027** | C1 |  | POP B | Pop BC from stack |
| **F028** | E1 |  | POP H | Pop HL from stack |
| **F029** | C9 |  | RET | Return from subroutine |
| **F02A** | F5 | SWAP | PUSH PSW | Store AF into stack |
| **F02B** | C5 |  | PUSH B | Push BC into stack |
| **F02C** | 1A |  | LDAX D | A = get data from location pointed by DE |
| **F02D** | 47 |  | MOV B,A | Copy A to B |
| **F02E** | 7E |  | MOV A,M | get data from location pointed by HL |
| **F02F** | 12 |  | STAX D | Store A content into memory pointed by DE |
| **F030** | 70 |  | MOV M,B | Store B content to memory pointed by HL |
| **F031** | C1 |  | POP B | Pop BC from Stack |
| **F032** | F1 |  | POP PSW | Pop AF from stack |
| **F033** | C9 |  | RET | Return from swap |