Python Programs Without Using Built-in Methods — Extended Master Reference

LIST OPERATIONS

```
1. Append element (without append)
lst = [1, 2, 3]
x = 4
lst = lst + [x]

2. Extend two lists (without extend)
a = [1, 2]
b = [3, 4]
for i in b:
    a = a + [i]

3. Insert at position (without insert)
lst = [10, 20, 30]
pos, val = 1, 99
new = []
for i in range(len(lst)):
    if i == pos:
        new = new + [val]
    new = new + [lst[i]]

4. Remove element (without remove)
lst = [2, 3, 4, 3]
val = 3
new = []
removed = False
for i in lst:
    if i == val and not removed:
        removed = True
    else:
        new = new + [i]

5. Reverse list (without reverse)
lst = [1, 2, 3]
rev = []
for i in range(len(lst)-1, -1, -1):
    rev = rev + [lst[i]]

6. Find length (without len)
lst = [10, 20, 30]
count = 0
for _ in lst:
    count += 1
```

STRING OPERATIONS

```
1. String length (without len)
s = "hello"
count = 0
for _ in s:
    count += 1

2. Reverse string
s = "python"
rev = ""
for i in range(len(s)-1, -1, -1):
    rev += s[i]

3. Count vowels
s = "education"
vowels = "aeiouAEIOU"
count = 0
```

```
for ch in s:
    for v in vowels:
        if ch == v:
            count += 1

4. Convert lowercase to uppercase (without upper)
s = "abcD"
res = ""
for ch in s:
    if 'a' <= ch <= 'z':
        res += chr(ord(ch) - 32)
    else:
        res += ch

5. Find substring (without find)
s, sub = "hello world", "wor"
found = False
for i in range(len(s) - len(sub) + 1):
    match = True
    for j in range(len(sub)):
        if s[i + j] != sub[j]:
            match = False
            break
    if match:
        found = True
```

MATH & ALGORITHMS

```
1. Sum of list (without sum)
arr = [5, 10, 15]
total = 0
for n in arr:
    total += n

2. Max number (without max)
arr = [4, 9, 1]
m = arr[0]
for x in arr:
    if x > m:
        m = x

3. Min number (without min)
arr = [4, 9, 1]
m = arr[0]
for x in arr:
    if x < m:
        m = x

4. Check prime number
n = 17
isPrime = True
if n < 2:
    isPrime = False
for i in range(2, n):
    if n % i == 0:
        isPrime = False
        break

5. Factorial (without math)
n = 5
fact = 1
for i in range(1, n+1):
    fact *= i

6. Fibonacci (loop)
n = 7
a, b = 0, 1
for _ in range(n):
    print(a)
    a, b = b, a + b
```

SEARCHING & SORTING

```
1. Linear search
arr = [4, 6, 8, 2]
key = 8
found = False
for x in arr:
    if x == key:
        found = True
        break

2. Binary search (sorted list)
arr = [2, 4, 6, 8, 10]
low, high = 0, len(arr)-1
key = 8
while low <= high:
    mid = (low + high) // 2
    if arr[mid] == key:
        print("Found")
        break
    elif arr[mid] > key:
        high = mid - 1
    else:
        low = mid + 1

3. Bubble sort
arr = [4, 2, 7, 1]
for i in range(len(arr)):
    for j in range(0, len(arr)-i-1):
        if arr[j] > arr[j+1]:
            arr[j], arr[j+1] = arr[j+1], arr[j]
```

DICTIONARY WITHOUT BUILT-INS

```
1. Create dictionary manually
keys = ["a", "b"]
values = [1, 2]
d = {}
for i in range(len(keys)):
    d[keys[i]] = values[i]

2. Search key-value
k = "b"
found = False
for key in d:
    if key == k:
        found = True

3. Delete key (without pop)
remove = "a"
new = {}
for key in d:
    if key != remove:
        new[key] = d[key]
```

SET OPERATIONS WITHOUT SET()

```
arr1 = [1, 2, 3]
arr2 = [3, 4, 5]

1. Union
union = arr1[:]
for x in arr2:
    exists = False
    for y in union:
        if x == y:
            exists = True
            break
    if not exists:
        union = union + [x]

2. Intersection
inter = []
for x in arr1:
    for y in arr2:
        if x == y:
            inter = inter + [x]

3. Difference
diff = []
for x in arr1:
    exists = False
    for y in arr2:
        if x == y:
            exists = True
            break
    if not exists:
        diff = diff + [x]
```

OOP WITHOUT USING BUILT-IN HELPERS

```
class Student:
    def __init__(self, name, marks):
        self.name = name
        self.marks = marks

    def average(self):
        total = 0
        for m in self.marks:
            total += m
        return total / len(self.marks)

s = Student("John", [80, 90, 70])
print(s.average())
```