ТК-31 Петрів Владислав

Структури даних і алгоритми

Задачі 1

1. **Робота з одновимірними масивами.** Для заданого масиву цілих чисел (довжина > 100, генеровані випадковим чином) знайти мінімальне значення, максимальне значення, впорядкувати масив (за спаданням, за зростанням).

Вивід:

Enter array length: 120

Array: 953 811 728 370 779 267 786 589 232 653 157 223 405 331 3 372 966 943 872 553 10 785 696 440 970 30 164 337 670 897 902 226 243 446 993 737 352 610 131 358 3 591 189 759 517 871 827 29 495 485 974 838 341 805 668 26 106 780 439 693 849 184 578 904 449 37 202 519 833 610 767 209 921 480 119 387 905 551 565 422 707 553 254 302 855 814 386 168 90 889 947 288 80 469 278 942 623 627 597 365 515 349 652 902 193 221 804 690 642 189 409 424 948 403 996 89 937 760 884 785

Min: 3

Max: 996

Sorting by desc: 996 993 974 970 966 953 948 947 943 942 937 921 905 904 902 902 897 889 884 872 871 855 849 838 833 827 814 811 805 804 786 785 785 780 779 767 760 759 737 728 707 696 693 690 670 668 653 652 642 627 623 610 610 597 591 589 578 565 553 553 551 519 517 515 495 485 480 469 449 446 440 439 424 422 409 405 403 387 386 372 370 365 358 352 349 341 337 331 302 288 278 267 254 243 232 226 223 221 209 202 193 189 189 184 168 164 157 131 119 106 90 89 80 37 30 29 26 10 3 3

Sorting by asc: 3 3 10 26 29 30 37 80 89 90 106 119 131 157 164 168 184 189 189 193 202 209 221 223 226 232 243 254 267 278 288 302 331 337 341 349 352 358 365 370 372 386 387 403 405 409 422 424 439 440 446 449 469 480 485 495 515 517 519 551 553 553 565 578 589 591 597 610 610 623 627 642 652 653 668 670 690 693 696 707 728 737 759 760 767 779 780 785 785 786 804 805 811 814 827 833 838 849 855 871 872 884 889 897 902 902 904 905 921 937 942 943 947 948 953 966 970 974 993 996

ТК-31 Петрів Владислав

Task A: 996 974 966 948 943 937 905 902 897 884 871 849 833 814 805 786 785 779 760 737 707 693 670 653 642 623 610 591 578 553 551 517 495 480 449 440 424 409 403 386 370 358 349 337 302 278 254 232 223 209 193 189 168 157 119 90 80 30 26 3 3 10 29 37 89 106 131 164 184 189 202 221 226 243 267 288 331 341 352 365 372 387 405 422 439 446 469 485 515 519 553 565 589 597 610 627 652 668 690 696 728 759 767 780 785 804 811 827 838 855 872 889 902 904 921 942 947 953 970 993

Task B: 996 3 993 3 974 10 970 26 966 29 953 30 948 37 947 80 943 89 942 90 937 106 921 119 905 131 904 157 902 164 902 168 897 184 889 189 884 189 872 193 871 202 855 209 849 221 838 223 833 226 827 232 814 243 811 254 805 267 804 278 786 288 785 302 785 331 780 337 779 341 767 349 760 352 759 358 737 365 728 370 707 372 696 386 693 387 690 403 670 405 668 409 653 422 652 424 642 439 627 440 623 446 610 449 610 469 597 480 591 485 589 495 578 515 565 517 553 519 553 551

Task C: 996 974 970 966 948 942 904 902 902 884 872 838 814 804 786 780 760 728 696 690 670 668 652 642 610 610 578 480 446 440 424 422 386 372 370 358 352 302 288 278 254 232 226 202 184 168 164 106 90 80 30 26 10 3 3 29 37 89 119 131 157 189 189 193 209 221 223 243 267 331 337 341 349 365 387 403 405 409 439 449 469 485 495 515 517 519 551 553 553 565 589 591 597 623 627 653 693 707 737 759 767 779 785 785 805 811 827 833 849 855 871 889 897 905 921 937 943 947 953 993

Генерація масиву:

```cpp
vector<int> CreateArray(const int& n) {

    std::random_device rd; // отримуємо випадкове число з компютера
    std::mt19937 gen(rd()); // сід генератора
    std::uniform_int_distribution<> distr(0, 1000); // границя випадкових чисел

    vector<int> arr;

    for (int i = 0; i < n; ++i) {
        arr.push_back(distr(gen));
    }
    return arr;
}
```

Макс, мін значення, сортування

```cpp
auto minNum = min_element(v.begin(), v.end());
auto maxNum = max_element(v.begin(), v.end());
```

```cpp
/*Desc*/ sort(v.begin(), v.end(), greater<int>());
```

```cpp
/*Asc*/  sort(v.begin(), v.end());
```

ТК-31 Петрів Владислав

А) в центрі - мінімальне (максимальне) значення, додаємо зліва – справа елементи в порядку зростання (спадання).

```cpp
vector<int> task1A(vector<int>& v) {
    vector<int> res;
    bool insertInBegin = false;
    for (int& i : v) {
        if (insertInBegin) {
            res.insert(res.begin(), i);
        }
        else {
            res.push_back(i);
        }
        insertInBegin = !insertInBegin;
    }
    return res;
}
```

Б) сформувати масив – максимальне, мінімальне, максимальне (з тих, що залишились), мінімальне, … Вивести масив.

```cpp
vector<int> task1C(vector<int>& v) {
    vector<int> desc, asc, res;
    asc = v;
    sort(v.begin(), v.end(), greater<int>());
    desc = v;
    for (int i = 0; i < desc.size(); ++i) {
        if (desc[i] % 2 == 0) {
            res.push_back(desc[i]);
        }
    }
    for (int i = 0; i < asc.size(); ++i) {
        if (asc[i] % 2 == 1) {
            res.push_back(asc[i]);
        }
    }
    return res;
}
```

В) сформувати масив – елементи з парними номерами спадають, потів – елементи з непарними номерами зростають. Вивести масив.

```cpp
vector<int> task1B(vector<int>& v) {
    vector<int> res;
    for (int i = 0; i < v.size() / 2 + 1 && res.size() < v.size(); ++i) {
        res.push_back(v[v.size() - i - 1]);
        if (res.size() < v.size()) {
            res.push_back(v[i]);
        }
    }
    return res;
}
```

ТК-31 Петрів Владислав

2) Робота з матрицями. Для заданого масиву цілих чисел (довжина > 10*10, генеровані випадковим чином) знайти мінімальне значення, максимальне значення, впорядкувати масив за правилом:

Вивід:

Enter num of raws: 15

Enter num of columns: 15

754 625 626 34 924 378 213 643 420 589 48 935 782 743 929

525 647 560 987 352 73 1 644 539 529 928 24 57 274 734

7 486 478 993 205 108 306 418 78 901 577 243 205 219 790

570 675 276 670 765 69 144 351 932 369 317 680 420 518 992

969 294 279 81 602 736 889 641 116 550 179 360 864 858 483

654 479 621 978 725 293 293 156 586 9 176 669 766 775 524

348 292 843 565 898 26 758 895 381 996 297 919 723 348 279

920 196 479 532 224 793 2 737 568 40 902 440 703 315 688

6 34 646 715 939 2 943 191 696 65 227 313 274 668 999

311 543 905 428 33 609 730 393 699 675 111 629 163 83 602

252 788 1 177 664 209 590 559 260 365 491 263 822 300 310

530 79 16 544 294 202 594 213 214 551 987 572 996 23 823

638 879 359 359 731 422 697 23 524 207 268 45 120 680 975

974 705 964 52 390 900 631 192 504 768 717 160 875 836 235

179 672 282 289 427 320 178 499 947 317 534 408 122 213 928

Min: 1

Max: 999

ТК-31 Петрів Владислав

Task 2 A:

1 1 2 2 6 7 9 16 23 23 24 26 33 34 34

40 45 48 52 57 65 69 73 78 79 81 83 108 111 116

120 122 144 156 160 163 176 177 178 179 179 191 192 196 202

205 205 207 209 213 213 213 214 219 224 227 235 243 252 260

263 268 274 274 276 279 279 282 289 292 293 293 294 294 297

300 306 310 311 313 315 317 317 320 348 348 351 352 359 359

360 365 369 378 381 390 393 408 418 420 420 422 427 428 440

478 479 479 483 486 491 499 504 518 524 524 525 529 530 532

534 539 543 544 550 551 559 560 565 568 570 572 577 586 589

590 594 602 602 609 621 625 626 629 631 638 641 643 644 646

647 654 664 668 669 670 672 675 675 680 680 688 696 697 699

703 705 715 717 723 725 730 731 734 736 737 743 754 758 765

766 768 775 782 788 790 793 822 823 836 843 858 864 875 879

889 895 898 900 901 902 905 919 920 924 928 928 929 932 935

939 943 947 964 969 974 975 978 987 987 992 993 996 996 999

Task 2 B:

1 40 120 205 263 300 360 478 534 590 647 703 766 889 939

1 45 122 205 268 306 365 479 539 594 654 705 768 895 943

2 48 144 207 274 310 369 479 543 602 664 715 775 898 947

2 52 156 209 274 311 378 483 544 602 668 717 782 900 964

6 57 160 213 276 313 381 486 550 609 669 723 788 901 969

7 65 163 213 279 315 390 491 551 621 670 725 790 902 974

9 69 176 213 279 317 393 499 559 625 672 730 793 905 975

16 73 177 214 282 317 408 504 560 626 675 731 822 919 978

23 78 178 219 289 320 418 518 565 629 675 734 823 920 987

23 79 179 224 292 348 420 524 568 631 680 736 836 924 987

24 81 179 227 293 348 420 524 570 638 680 737 843 928 992

ТК-31 Петрів Владислав

26 83 191 235 293 351 422 525 572 641 688 743 858 928 993

33 108 192 243 294 352 427 529 577 643 696 754 864 929 996

34 111 196 252 294 359 428 530 586 644 697 758 875 932 996

34 116 202 260 297 359 440 532 589 646 699 765 879 935 999

Task 2 C:

1 1 2 2 6 7 9 16 23 23 24 26 33 34 34

227 235 243 252 260 263 268 274 274 276 279 279 282 289 40

224 428 440 478 479 479 483 486 491 499 504 518 524 292 45

219 427 629 631 638 641 643 644 646 647 654 664 524 293 48

214 422 626 737 743 754 758 765 766 768 775 668 525 293 52

213 420 625 736 901 902 905 919 920 924 782 669 529 294 57

213 420 621 734 900 974 975 978 987 928 788 670 530 294 65

213 418 609 731 898 969 996 999 987 928 790 672 532 297 69

209 408 602 730 895 964 996 993 992 929 793 675 534 300 73

207 393 602 725 889 947 943 939 935 932 822 675 539 306 78

205 390 594 723 879 875 864 858 843 836 823 680 543 310 79

205 381 590 717 715 705 703 699 697 696 688 680 544 311 81

202 378 589 586 577 572 570 568 565 560 559 551 550 313 83

196 369 365 360 359 359 352 351 348 348 320 317 317 315 108

192 191 179 179 178 177 176 163 160 156 144 122 120 116 111

Task 2 D:

999 723 725 730 731 734 736 737 743 754 758 765 766 768 775

996 717 539 543 544 550 551 559 560 565 568 570 572 577 782

996 715 534 317 317 320 348 348 351 352 359 359 360 586 788

993 705 532 315 213 213 213 214 219 224 227 235 365 589 790

992 703 530 313 209 81 83 108 111 116 120 243 369 590 793

987 699 529 311 207 79 23 24 26 33 122 252 378 594 822

ТК-31 Петрів Владислав

987 697 525 310 205 78 23 1 2 34 144 260 381 602 823

978 696 524 306 205 73 16 1 2 34 156 263 390 602 836

975 688 524 300 202 69 9 7 6 40 160 268 393 609 843

974 680 518 297 196 65 57 52 48 45 163 274 408 621 858

969 680 504 294 192 191 179 179 178 177 176 274 418 625 864

964 675 499 294 293 293 292 289 282 279 279 276 420 626 875

947 675 491 486 483 479 479 478 440 428 427 422 420 629 879

943 672 670 669 668 664 654 647 646 644 643 641 638 631 889

939 935 932 929 928 928 924 920 919 905 902 901 900 898 895

Task 2 E:

1 1 2 2 6 7 9 16 23 23 24 26 33 34 34

116 111 108 83 81 79 78 73 69 65 57 52 48 45 40

120 122 144 156 160 163 176 177 178 179 179 191 192 196 202

260 252 243 235 227 224 219 214 213 213 213 209 207 205 205

263 268 274 274 276 279 279 282 289 292 293 293 294 294 297

359 359 352 351 348 348 320 317 317 315 313 311 310 306 300

360 365 369 378 381 390 393 408 418 420 420 422 427 428 440

532 530 529 525 524 524 518 504 499 491 486 483 479 479 478

534 539 543 544 550 551 559 560 565 568 570 572 577 586 589

646 644 643 641 638 631 629 626 625 621 609 602 602 594 590

647 654 664 668 669 670 672 675 675 680 680 688 696 697 699

765 758 754 743 737 736 734 731 730 725 723 717 715 705 703

766 768 775 782 788 790 793 822 823 836 843 858 864 875 879

935 932 929 928 928 924 920 919 905 902 901 900 898 895 889

939 943 947 964 969 974 975 978 987 987 992 993 996 996 999

Task 2 F:

1 1 252 260 263 268 530 532 534 539 758 765 766 768 939

ТК-31 Петрів Владислав

2 2 243 274 274 543 529 775 544 529 754 743 782 274 939

6 7 224 227 276 279 524 524 550 551 736 737 788 790 939

16 9 219 279 282 559 518 793 560 369 734 731 822 518 939

23 23 213 213 289 292 491 499 565 568 725 730 823 836 939

26 24 213 293 293 570 486 843 572 9 723 717 858 775 939

33 34 205 207 294 294 479 479 577 586 705 715 864 875 939

40 34 205 297 300 589 478 879 590 40 703 699 889 315 939

45 48 192 196 306 310 427 428 594 602 696 697 895 898 939

57 52 191 311 313 602 422 900 609 675 688 680 901 83 939

65 69 178 179 315 317 418 420 621 625 675 680 902 905 939

78 73 177 317 320 626 408 919 629 551 675 672 920 23 939

79 81 160 163 348 348 381 390 631 638 669 670 924 928 939

108 83 156 351 352 641 378 928 643 768 668 664 929 836 939

111 116 120 122 359 359 360 365 644 646 647 654 932 935 939

Генерація масиву:

```cpp
vector<vector<int>> CreateArray(const int& n1 /*рядки*/, const int& n2 /*стовбці*/) {

    std::random_device rd; // отримуємо випадкове число з компютера
    std::mt19937 gen(rd()); // сід генератора
    std::uniform_int_distribution<> distr(0, 1000); // границя випадкових чисел

    vector<vector<int>> arr(n1);

    for (int i = 0; i < n1; ++i) {
        for (int j = 0; j < n2; ++j)
            arr[i].push_back(distr(gen));
    }
    return arr;
}
```

Максимальне та мінімальне значення:

```cpp
int Max(const vector<vector<int>>& vv, int xn, int yn) {
    int max = std::numeric_limits<int>::min();
    for (int x = 0; x < xn; x++) {
        for (int y = 0; y < yn; y++) {
            if (vv[x][y] > max) {
                max = vv[x][y];
            }
        }
    }
    return max;
}
```

ТК-31 Петрів Владислав

```cpp
int Min(const vector<vector<int>>& vv, int xn, int yn) {
    int min = std::numeric_limits<int>::max();
    for (int x = 0; x < xn; x++) {
        for (int y = 0; y < yn; y++) {
            if (vv[x][y] < min) {
                min = vv[x][y];
            }
        }
    }
    return min;
}
```

А)

```cpp
void task2A(vector<vector<int>> vv, int n1, int n2) {
    vector<int> v;
    for (int i = 0; i < n1; ++i) {
        for (int j = 0; j < n2; ++j) {
            v.push_back(vv[i][j]);
        }
    }
    sort(v.begin(), v.end());

    int count = 0;
    for (int i = 0; i < n1; ++i) {
        for (int j = 0; j < n2; ++j) {
            vv[i][j] = v[count];
            count++;
        }
    }
    printVector(vv);
}
```

Б)

```cpp
void task2B(vector<vector<int>> vv, int n1, int n2) {
    vector<int> v;
    for (int i = 0; i < n1; ++i) {
        for (int j = 0; j < n2; ++j) {
            v.push_back(vv[i][j]);
        }
    }
    sort(v.begin(), v.end());
    int count = 0;
    for (int column = 0; column < n2; column++) {
        for (int raw = 0; raw < n1; raw++) {
            vv[raw][column] = v[count];
            count++;
        }
    }
    printVector(vv);
```

ТК-31 Петрів Владислав

В)

```
void task2C(vector<vector<int>> vv, int n1, int n2) {

    vector<int> v;
    for (int i = 0; i < n1; ++i) {
        for (int j = 0; j < n2; ++j) {
            v.push_back(vv[i][j]);
        }
    }
    sort(v.begin(), v.end());

    int count = 0;
    int left = 0;
    int right = n2 - 1;
    int up = 0;
    int down = n1 - 1;
    int x = 0;
    int y = 0;
    while (left < right + 1 || up < down + 1) {
        if (y == up && x < right) {
            vv[up][x] = v[count];
            if (x < right) {
                x++;
            }
            if (x == right) {
                up++;
            }
        }
```

```
        else if (x == right && y < down) {
            vv[y][right] = v[count];
            if (y < down) {
                y++;
            }
            if (y == down) {
                right--;
            }
        }
        else if (y == down && x > left) {
            vv[down][x] = v[count];
            if (x > left) {
                x--;
            }
            if (x == left) {
                down--;
            }
        }
        else if (x == left && y > up) {
            vv[y][left] = v[count];
            if (y > up) {
                y--;
            }
            if (y == up) {
                left++;
            }
        }
        else
        {
            vv[y][x] = v[count];
            break;
        }
        count++;
}
printVector(vv);
```

ТК-31 Петрів Владислав

Г)

```
void task2D(vector<vector<int>> vv, int n1, int n2) {
    vector<int> v;
    for (int i = 0; i < n1; ++i) {
        for (int j = 0; j < n2; ++j) {
            v.push_back(vv[i][j]);
        }
    }
    sort(v.begin(), v.end(), greater<int>());
    int count = 0;
    int left = 0;
    int right = n2 - 1;
    int up = 0;
    int down = n1 - 1;
    int x = 0;
    int y = 0;
    vector<int> res;
    while (left < right + 1 || up < down + 1) {
        if (y == up && x > left) {
            vv[up][x] = v[count];
            if (x > left) {
                x--;
            }
            if (x == left) {
                up++;
            }
        }
        else if (x == right && y > up) {
            vv[y][right] = v[count];
            if (y > up) {
                y--;
            }
            if (y == up) {
                right--;
            }
        }
        else if (y == down && x < right) {
            vv[down][x] = v[count];
            if (x < right) {
                x++;
            }
            if (x == right) {
                down--;
            }
        }
        else if (x == left && y < down) {
            vv[y][left] = v[count];
            if (y < down) {
                y++;
            }
            if (y == down) {
                left++;
            }
        }
        else
        {
            vv[y][x] = v[count];
            break;
        }
        count++;
    }
    printVector(vv);
}
```

ТК-31 Петрів Владислав

Д)

```
void task2E(vector<vector<int>> vv, int n1, int n2)

    vector<int> v;
    for (int i = 0; i < n1; ++i) {
        for (int j = 0; j < n2; ++j) {
            v.push_back(vv[i][j]);
        }
    }
    sort(v.begin(), v.end());

    int count = 0;

    bool printLeftToRight = true;
    for (int i = 0; i < n1; ++i) {
        if (printLeftToRight)
        {
            for (int j = 0; j < n2; j++) {
                vv[i][j] = v[count];
                count++;
            }
        }
        else {
            for (int j = n2 - 1; j >= 0; j--) {
                vv[i][j] = v[count];
                count++;
            }
        }
        printLeftToRight = !printLeftToRight;
    }
    printVector(vv);
}
```

Е)

```
void task2F(vector<vector<int>> vv, int n1, int n2) {

    vector<int> v;
    for (int i = 0; i < n1; ++i) {
        for (int j = 0; j < n2; ++j) {
            v.push_back(vv[i][j]);
        }
    }
    sort(v.begin(), v.end());

    int count = 0;

    bool leftToRight = true;
    bool upToDown = true;
    for (int doubleColumn = 0; doubleColumn < n2 / 2; ++doubleColumn) {
        leftToRight = true;
        if (upToDown) {
            for (int raw = 0; raw < n1; raw++) {
                if (leftToRight) {
                    vv[raw][doubleColumn * 2] = v[count];
                    count++;
                    vv[raw][doubleColumn * 2 + 1] = v[count];
                    count++;
                }
                else {
                    vv[raw][doubleColumn + 1] = v[count];
                    count++;
                    vv[raw][doubleColumn * 2] = v[count];
                    count++;
                }
                leftToRight = !leftToRight;
            }
        }
```

```
        else {
            for (int raw = n1 - 1; raw >= 0; raw--) {
                if (leftToRight) {
                    vv[raw][doubleColumn * 2] = v[count];
                    count++;
                    vv[raw][doubleColumn * 2 + 1] = v[count];
                    count++;
                }
                else {
                    vv[raw][doubleColumn * 2 + 1] = v[count];
                    count++;
                    vv[raw][doubleColumn * 2] = v[count];
                    count++;
                }
                leftToRight = !leftToRight;
            }
        }
        upToDown = !upToDown;
    }
    if (n2 % 2 == 1) {
        if (upToDown) {
            for (int i = 0; i < n1; i++) {
                vv[i][n2 - 1] = v[count];
                count++;
            }
        }
        else {
            for (int i = n1 - 1; i >= 0; i--) {
                vv[i][n2 - 1] = v[count];
            }
        }
    }
    printVector(vv);
}
```

3) Трьохвимірний масив. Для заданого масиву цілих чисел (довжина > 5*5*, генеровані випадковим чином) знайти мінімальне значення, максимальне значення, впорядкувати масив за зростанням по осі OX, потім по осі OY, OZ. Вивести масив.

Вивід:

Enter z:

7

Enter y:

7

Enter x:

7

861 525 817 516 583 961 948

205 7 492 696 606 780 526

85 783 246 10 582 35 149

ТК-31 Петрів Владислав

822 386 294 655 116 548 800

295 154 155 94 936 688 931

806 420 119 191 662 509 966

101 441 625 885 836 494 383

837 347 563 502 661 323 243

563 973 235 89 962 294 386

925 403 125 233 887 361 395

656 143 803 925 342 456 118

244 154 940 948 166 746 683

113 728 904 831 231 630 302

517 11 148 594 607 478 704

680 360 860 199 332 707 135

838 79 825 627 790 761 399

61 851 418 612 378 2 563

120 968 501 571 780 896 497

236 510 525 776 264 324 505

646 950 94 223 691 832 533

146 113 143 968 362 335 819

467 756 903 817 61 118 694

787 546 324 703 672 438 705

126 916 782 640 887 11 66

951 831 499 22 497 310 951

250 585 839 797 714 435 554

ТК-31 Петрів Владислав

833 646 703 434 375 467 515

847 146 60 172 374 899 56

240 53 742 504 731 949 868

470 604 689 503 354 298 408

769 650 202 886 240 922 917

663 443 907 976 592 591 549

500 344 475 547 330 848 348

53 644 731 700 751 976 602

710 781 695 412 670 871 220

689 789 727 477 447 721 690

451 936 400 116 408 605 444

702 584 280 397 453 753 611

317 216 724 377 173 842 731

355 605 558 54 376 784 409

828 181 908 99 771 284 144

954 570 708 824 729 612 192

921 484 979 353 706 983 631

39 47 130 718 45 378 297

26 972 777 987 130 831 701

95 495 72 599 284 320 276

641 300 996 283 869 967 387

637 26 206 400 837 887 213

920 984 107 796 555 438 441

ТК-31 Петрів Владислав

Task 3 Max: 996

Task 3 Min: 2

Task 3 Sort X:

516 525 583 817 861 948 961

7 205 492 526 606 696 780

10 35 85 149 246 582 783

116 294 386 548 655 800 822

94 154 155 295 688 931 936

119 191 420 509 662 806 966

101 383 441 494 625 836 885

243 323 347 502 563 661 837

89 235 294 386 563 962 973

125 233 361 395 403 887 925

118 143 342 456 656 803 925

154 166 244 683 746 940 948

113 231 302 630 728 831 904

11 148 478 517 594 607 704

135 199 332 360 680 707 860

79 399 627 761 790 825 838

2 61 378 418 563 612 851

120 497 501 571 780 896 968

236 264 324 505 510 525 776

ТК-31 Петрів Владислав

94 223 533 646 691 832 950

113 143 146 335 362 819 968

61 118 467 694 756 817 903

324 438 546 672 703 705 787

11 66 126 640 782 887 916

22 310 497 499 831 951 951

250 435 554 585 714 797 839

375 434 467 515 646 703 833

56 60 146 172 374 847 899

53 240 504 731 742 868 949

298 354 408 470 503 604 689

202 240 650 769 886 917 922

443 549 591 592 663 907 976

330 344 348 475 500 547 848

53 602 644 700 731 751 976

220 412 670 695 710 781 871

447 477 689 690 721 727 789

116 400 408 444 451 605 936

280 397 453 584 611 702 753

173 216 317 377 724 731 842

54 355 376 409 558 605 784

99 144 181 284 771 828 908

192 570 612 708 729 824 954

ТК-31 Петрів Владислав

353 484 631 706 921 979 983

39 45 47 130 297 378 718

26 130 701 777 831 972 987

72 95 276 284 320 495 599

283 300 387 641 869 967 996

26 206 213 400 637 837 887

107 438 441 555 796 920 984

Task 3 Sort Y:

7 35 85 149 246 582 780

10 154 155 295 606 696 783

94 191 386 494 625 800 822

101 205 420 509 655 806 885

116 294 441 526 662 836 936

119 383 492 548 688 931 961

516 525 583 817 861 948 966

11 143 244 386 403 607 704

89 148 294 395 563 661 837

113 166 302 456 563 803 904

118 231 342 502 594 831 925

125 233 347 517 656 887 925

154 235 361 630 728 940 948

243 323 478 683 746 962 973

ТК-31 Петрів Владислав

2 61 146 335 362 525 776

79 143 324 360 510 612 838

94 199 332 418 563 707 851

113 223 378 505 680 819 860

120 264 501 571 691 825 950

135 399 533 646 780 832 968

236 497 627 761 790 896 968

11 60 126 172 374 703 787

22 66 146 499 646 705 833

56 118 467 515 703 797 839

61 310 467 585 714 817 899

250 434 497 640 756 847 903

324 435 546 672 782 887 916

375 438 554 694 831 951 951

53 240 348 470 500 547 689

53 240 408 475 503 604 848

202 344 504 592 663 751 871

220 354 591 695 710 781 922

298 412 644 700 731 868 949

330 549 650 731 742 907 976

443 602 670 769 886 917 976

54 144 181 284 451 605 753

ТК-31 Петрів Владислав

99 216 317 377 558 605 784

116 355 376 409 611 702 789

173 397 408 444 721 727 842

192 400 453 584 724 731 908

280 477 612 690 729 824 936

447 570 689 708 771 828 954

26 45 47 130 297 378 599

26 95 213 284 320 495 718

39 130 276 400 637 837 887

72 206 387 555 796 920 983

107 300 441 641 831 967 984

283 438 631 706 869 972 987

353 484 701 777 921 979 996

Task 3 Sort Z:

2 35 47 130 246 378 599

10 66 146 284 320 495 718

39 118 276 400 563 702 789

61 205 342 444 594 727 842

107 233 347 517 656 731 903

119 235 361 548 688 824 916

236 323 478 683 746 828 951

7 45 85 149 297 525 689

22 95 155 295 503 604 783

ТК-31 Петрів Владислав

56 130 302 409 563 707 822

72 206 378 502 655 781 860

116 264 441 526 662 825 908

135 383 492 630 728 832 936

243 438 554 694 771 896 954

11 60 126 172 362 547 704

26 143 213 360 510 605 784

94 166 332 418 611 751 839

101 223 387 505 680 806 885

120 294 441 571 691 836 925

154 399 533 646 729 887 948

353 484 583 708 790 917 966

11 61 146 284 374 582 753

53 148 294 377 558 612 833

94 191 376 456 625 797 851

113 231 408 509 710 817 899

125 300 453 584 724 847 936

280 435 546 672 742 907 961

375 497 627 761 831 948 968

26 143 181 335 403 605 776

79 154 317 395 563 661 837

113 199 386 494 637 800 871

118 310 420 555 714 819 922

ТК-31 Петрів Владислав

192 400 497 640 731 868 949

283 438 612 690 780 931 968

443 525 670 769 861 951 973

53 144 244 386 451 607 780

89 216 324 475 606 696 838

116 344 467 515 663 803 887

173 354 467 585 721 831 925

250 412 501 641 756 887 950

324 477 631 706 782 940 976

447 570 689 777 886 962 976

54 240 348 470 500 703 787

99 240 408 499 646 705 848

202 355 504 592 703 837 904

220 397 591 695 796 920 983

298 434 644 700 831 967 984

330 549 650 731 869 972 987

516 602 701 817 921 979 996

Генерація масиву:

```
vector<vector<vector<int>>>  CreateArray(int xn, int yn, int zn) {

    std::random_device rd; // отримуємо випадкове число з компютера
    std::mt19937 gen(rd()); // сід генератора
    std::uniform_int_distribution<> distr(0, 1000); // границя випадкових чисел

    vector<vector<vector<int>>> arr(zn);

    for (int i = 0; i < zn; ++i) {
        arr[i].resize(yn);
    }


    for (int i = 0; i < zn; ++i) {
        for (int j = 0; j < yn; ++j)
            for (int k = 0; k < xn; ++k) {
                arr[i][j].push_back(distr(gen));
            }
    }
    return arr;
}
```

ТК-31 Петрів Владислав

Макс, мін:

```
int Max(const vector<vector<vector<int>>>& vvv, int xn, int yn, int zn) {
    int max = std::numeric_limits<int>::min();
    for (int z = 0; z < zn; z++) {
        for (int y = 0; y < yn; y++) {
            for (int x = 0; x < xn; x++) {
                if (vvv[z][y][x] > max) {
                    max = vvv[z][y][x];
                }
            }
        }
    }
    return max;
}

int Min(const vector<vector<vector<int>>>& vvv, int xn, int yn, int zn) {
    int min = std::numeric_limits<int>::max();
    for (int z = 0; z < zn; z++) {
        for (int y = 0; y < yn; y++) {
            for (int x = 0; x < xn; x++) {
                if (vvv[z][y][x] < min) {
                    min = vvv[z][y][x];
                }
            }
        }
    }
    return min;
}
```

А)

```
void Task3SortX(vector<vector<vector<int>>>& vvv, int xn, int yn, int zn) {
    for (int z = 0; z < zn; z++) {
        for (int y = 0; y < yn; ++y) {
            sort(vvv[z][y].begin(), vvv[z][y].end());
        }
    }
}
```

Б)

```
void Task3SortY(vector<vector<vector<int>>>& vvv, int xn, int yn, int zn) {
    vector<int> v;
    for (int z = 0; z < zn; z++) {
        for (int x = 0; x < xn; x++) {
            for (int y = 0; y < yn; y++) {
                v.push_back(vvv[z][y][x]);
            }
            sort(v.begin(), v.end());
            for (int y = 0; y < yn; y++) {
                vvv[z][y][x] = v[y];
            }
            v.clear();
        }
    }
}
```

ТК-31 Петрів Владислав

В)

```
void Task3SortZ(vector<vector<vector<int>>>& vvv, int xn, int yn, int zn) {
    vector<int> v;
    for (int y = 0; y < yn; y++) {
        for (int x = 0; x < xn; x++) {
            for (int z = 0; z < zn; z++) {
                v.push_back(vvv[z][y][x]);
            }
            sort(v.begin(), v.end());
            for (int z = 0; z < zn; z++) {
                vvv[z][y][x] = v[z];
            }
            v.clear();
        }
    }

}
```

ТК-31 Петрів Владислав

4) **Операції над множинами.** Реалізувати операції над множинами, що задані у вигляді масивів. Операції – об’єднання, перетин, доповнення, різниця, симетрична різниця.

Вивід:

Task 4

Enter v1 length: 10

Enter v2 length: 15

v1 :

385 789 450 581 322 805 430 24 410 766

v2 :

313 998 497 272 288 709 729 665 883 131 509 538 761 330 372

Task 4 Union

385 789 450 581 322 805 430 24 410 766 313 998 497 272 288 709 729 665 883 131 509 538 761 330 372

Task 4 Cross

Task 4 Diff

385 789 450 581 322 805 430 24 410 766

Task 4 Symmetric difference

385 789 450 581 322 805 430 24 410 766 313 998 497 272 288 709 729 665 883 131 509 538 761 330 372

ТК-31 Петрів Владислав

Об’єднання)

```
vector<int> Task4Union(const vector<int>& v1, const vector<int>& v2) {
    vector<int> res = v1;
    vector<int> temp = v1;

    for (auto& el : v2) {
        auto it = find(temp.begin(), temp.end(), el);
        if (it != temp.end()) {
            temp.erase(it);
        }
        else {
            res.push_back(el);
        }
    }
    return res;
```

Перетин)

```
vector<int> Task4Cross(const vector<int>& v1, const vector<int>& v2) {
    vector<int> temp = v2;
    vector<int> res;
    for (const auto& el : v1) {
        auto it = find(temp.begin(), temp.end(), el);
        if (it != temp.end()) {
            res.push_back(el);
            temp.erase(it);
        }
    }
    return res;
}
```

Різниця)

```
vector<int> Task4Diff(const vector<int>& v1, const vector<int>& v2) {
    vector<int> cross = Task4Cross(v1, v2);
    vector<int> res;
    for (const auto& el : v1) {
        auto it = find(cross.begin(), cross.end(), el);
        if (it == cross.end()) {
            res.push_back(el);
        }
        else {
            cross.erase(it);
        }
    }
    return res;
}
```

ТК-31 Петрів Владислав

Симетрична різниця)

```
vector<int> Task4DiffSym(const vector<int>& v1, const vector<int>& v2) {
    vector<int> union_ = Task4Union(v1, v2);
    vector<int> cross = Task4Cross(v1, v2);
    vector<int> res;

    for (const auto& el : union_) {
        auto it = find(cross.begin(), cross.end(), el);
        if (it == cross.end()) {
            res.push_back(el);
        }
        else {
            cross.erase(it);
        }
    }
    return res;
}
```