Laporan Resmi
Praktikum Algoritma dan Struktur Data
Insertion and Selection Sort

Dr. Tita Karlita S.Kom, M.Kom

Nama : **Marits Ikmal Yasin**

Kelas : **1D4 IT B**

NRP : **3121600047**

1. Buatlah program yang mengimplementasikan algoritma selection sort, untuk mengurutkan 10 buah elemen array bertipe integer. Tampilkan hasil pengurutan data dalam urutan ascending. Tampilkan jumlah perbandingan dan jumlah penukaran elemen. Gunakan data berikut ini dalam 4 kali running program untuk melihat efek dari distribusi elemen terhadap jumlah penukaran dan jumlah perbandingan.
    a) array 1: 3, 10, 4, 6, 8, 9, 7, 2, 1, 5 -- data awal, semua elemen terdistribusi acak
    b) array 2: 1, 2, 3, 4, 5, 9, 7, 6, 8, 10-- data awal, 5 elemen pertama terurut descending, dan 5 elemen terakhir acak
    c) array 3: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 -- data awal, semua elemen terurut ascending
    d) array 4: 10, 9, 8, 7, 6, 5, 4, 3, 2, 1 -- data awal, semua elemen terurut descending

Kode :

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/time.h>
#include <unistd.h>

int perbandingan, penukaran;
void printArray(int arr[], int sz)
{
   for (int i = 0; i < sz; i++)
   {
      printf("%d ", arr[i]);
   }
   printf("\n");
}

void selectionSort(int arr[], int sz)
{
   for (int step = 0; step < sz - 1; step++)
   {
      usleep(1000);
      int min = step;
      for (int i = step + 1; i < sz; i++)
      {
         perbandingan++;
         if (arr[i] < arr[min])
         {
            min = i;
         }
      }
      if (arr[min] != arr[step])
         penukaran++;
      int temp = arr[min];
      arr[min] = arr[step];
      arr[step] = temp;
   }
}

int main()
{
   struct timeval start, stop;
   /*a*/int array[10] = {3, 10, 4, 6, 8, 9, 7, 2, 1, 5};
   /*b*/ int array[10] = {1, 2, 3, 4, 5, 9, 7, 6, 8, 10};
   /*c*/ int array[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
   /*d*/ int array[10] = {10, 9, 8, 7, 6, 5, 4, 3, 2, 1};
   int size = sizeof(array) / sizeof(array[0]);

   puts("Data Sebelum Sorting");
```

```
    printArray(array, size);
    gettimeofday(&start, NULL);
    selectionSort(array, size);
    gettimeofday(&stop, NULL);
    puts("Data Setelah Sorting");
    printArray(array, size);
    printf("Sorting %lu us\n", ((stop.tv_sec - start.tv_sec) * 1000000 + stop.tv_usec -
start.tv_usec));
    printf("Perbandingan = %d\n", perbandingan);
    printf("Penukaran = %d\n", penukaran);
    return 0;
}
```

Output :

a. int array[10] = {3, 10, 4, 6, 8, 9, 7, 2, 1, 5};

```
Data Sebelum Sorting
3 10 4 6 8 9 7 2 1 5
Data Setelah Sorting
1 2 3 4 5 6 7 8 9 10
Sorting 704018 us
Perbandingan = 45
Penukaran = 7
```

b. int array[10] = {1, 2, 3, 4, 5, 9, 7, 6, 8, 10};

```
Data Sebelum Sorting
1 2 3 4 5 9 7 6 8 10
Data Setelah Sorting
1 2 3 4 5 6 7 8 9 10
Sorting 697162 us
Perbandingan = 45
Penukaran = 2
```

c. int array[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};

```
Data Sebelum Sorting
1 2 3 4 5 6 7 8 9 10
Data Setelah Sorting
1 2 3 4 5 6 7 8 9 10
Sorting 697664 us
Perbandingan = 45
Penukaran = 0
```

d. int array[10] = {10, 9, 8, 7, 6, 5, 4, 3, 2, 1};

```
Data Sebelum Sorting
10 9 8 7 6 5 4 3 2 1
Data Setelah Sorting
1 2 3 4 5 6 7 8 9 10
Sorting 703715 us
Perbandingan = 45
Penukaran = 5
```

2. Buatlah program yang mengimplementasikan algoritma insertion sort, untuk mengurutkan 10 buah elemen array bertipe integer. Tampilkan hasil pengurutan data dalam urutan ascending. Tampilkan jumlah perbandingan dan jumlah penukaran elemen.

Gunakan data berikut ini dalam 4 kali running program untuk melihat efek dari distribusi elemen terhadap jumlah pergeseran, jumlah perbandingan, dan jumlah penyisipan.

a) array 1: 3, 10, 4, 6, 8, 9, 7, 2, 1, 5 -- data awal, semua elemen terdistribusi acak
b) array 2: 1, 2, 3, 4, 5, 9, 7, 6, 8, 10-- data awal, 5 elemen pertama terurut descending, dan 5 elemen terakhir acak
c) array 3: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 -- data awal, semua elemen terurut ascending
d) array 4: 10, 9, 8, 7, 6, 5, 4, 3, 2, 1 -- data awal, semua elemen terurut descending

Kode :

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/time.h>
#include <unistd.h>

int perbandingan, pergeseran, penyisipan;

void printArray(int arr[], int sz)
{
   for (int i = 0; i < sz; i++)
   {
      printf("%d ", arr[i]);
   }
   printf("\n");
}

void insertionSort(int arr[], int sz)
{
   for (int step = 1; step < sz; step++)
   {
      int key = arr[step];
      int back = step - 1;

      // best case comparison is n-1
      if (key > arr[back])
      {
         perbandingan++;
         usleep(1000);
      }
      while (key < arr[back] && back >= 0)
      {
         // worst case comparison is n(n-1)/2
         usleep(1000);
         pergeseran++;
         arr[back + 1] = arr[back];
         back--;
         perbandingan++;
      }
      if (arr[back + 1] != key)
         penyisipan++;
      arr[back + 1] = key;
   }
}

int main()
{
   struct timeval start, stop;
   /*a*/ int array[10] = {3, 10, 4, 6, 8, 9, 7, 2, 1, 5};
   /*b*/ int array[10] = {1, 2, 3, 4, 5, 9, 7, 6, 8, 10};
   /*c*/ int array[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
```

```
    /*d*/ int array[10] = {10, 9, 8, 7, 6, 5, 4, 3, 2, 1};
    int size = sizeof(array) / sizeof(array[0]);

    puts("Data Sebelum Sorting");
    printArray(array, size);
    gettimeofday(&start, NULL);
    insertionSort(array, size);
    gettimeofday(&stop, NULL);
    puts("Data Setelah Sorting");
    printArray(array, size);
    printf("Sorting %lu us\n", ((stop.tv_sec - start.tv_sec) * 1000000 + stop.tv_usec -
start.tv_usec));
    printf("Perbandingan = %d\n", perbandingan);
    printf("Pergeseran = %d\n", pergeseran);
    printf("Penyisipan = %d\n", penyisipan);
    return 0;
}
```

Output :

a. int array[10] = {3, 10, 4, 6, 8, 9, 7, 2, 1, 5};

```
Data Sebelum Sorting
3 10 4 6 8 9 7 2 1 5
Data Setelah Sorting
1 2 3 4 5 6 7 8 9 10
Sorting 425649 us
Perbandingan = 28
Pergeseran = 27
Penyisipan = 8
```

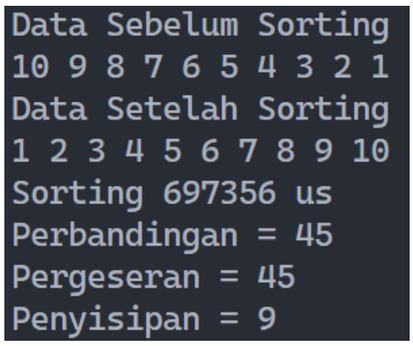
b. int array[10] = {1, 2, 3, 4, 5, 9, 7, 6, 8, 10};

```
Data Sebelum Sorting
1 2 3 4 5 9 7 6 8 10
Data Setelah Sorting
1 2 3 4 5 6 7 8 9 10
Sorting 188836 us
Perbandingan = 13
Pergeseran = 4
Penyisipan = 3
```

c. int array[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};

```
Data Sebelum Sorting
1 2 3 4 5 6 7 8 9 10
Data Setelah Sorting
1 2 3 4 5 6 7 8 9 10
Sorting 131365 us
Perbandingan = 9
Pergeseran = 0
Penyisipan = 0
```

d. int array[10] = {10, 9, 8, 7, 6, 5, 4, 3, 2, 1};



3. Buatlah program yang mengimplementasikan algoritma insertion dan selection sort, untuk mengurutkan 10 buah elemen array bertipe integer dengan menggunakan array berikut sebagai inisialisasi nilia awal array: 3, 10, 4, 6, 8, 9, 7, 2, 1, 5

   Kode :

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/time.h>
#include <unistd.h>

int perbandingan, penukaran;

void printArray(int arr[], int sz)
{
    for (int i = 0; i < sz; i++)
    {
        printf("%d ", arr[i]);
    }
    printf("\n");
}

// Ascending
void AscSelectionSort(int arr[], int sz)
{
    for (int step = 0; step < sz - 1; step++)
    {
        int min = step;
        for (int i = step + 1; i < sz; i++)
        {
            usleep(100);
            perbandingan++;
            if (arr[i] < arr[min])
                min = i;
        }
        if (arr[min] != arr[step])
            penukaran++;
        int temp = arr[min];
        arr[min] = arr[step];
        arr[step] = temp;
    }
}

// Descending
void DescSelectionSort(int arr[], int sz)
{
    for (int step = 0; step < sz - 1; step++)
    {
        int min = step;
```

```
        for (int i = step + 1; i < sz; i++)
        {
            usleep(100);
            perbandingan++;
            if (arr[i] > arr[min])
                min = i;
        }
        if (arr[min] != arr[step])
            penukaran++;
        int temp = arr[min];
        arr[min] = arr[step];
        arr[step] = temp;
    }
}

// Ascending
void AscInsertionSort(int arr[], int sz)
{
    for (int step = 1; step < sz; step++)
    {
        int key = arr[step];
        int back = step - 1;

        while (key < arr[back] && back >= 0)
        {
            usleep(100);
            arr[back + 1] = arr[back];
            back--;
        }
        if (key > arr[back])
        {
            perbandingan++;
            usleep(100);
        }
        if (arr[back + 1] != key)
            penukaran++;
        arr[back + 1] = key;
    }
}

// Descending
void DescInsertionSort(int arr[], int sz)
{
    for (int step = 1; step < sz; step++)
    {
        int key = arr[step];
        int back = step - 1;

        while (key > arr[back] && back >= 0)
        {
            usleep(100);
            arr[back + 1] = arr[back];
            back--;
        }
        if (key < arr[back])
        {
            perbandingan++;
            usleep(100);
```

```
        }
        if (arr[back + 1] != key)
            penukaran++;
        arr[back + 1] = key;
    }
}

void menu()
{
    puts("MENU METODE SORTING");
    puts("1. Insertion Sort");
    puts("2. Selection Sort");
    puts("3. Keluar");
}

void insertion()
{
    puts("MENU METODE SORTING");
    puts("1. Ascending");
    puts("2. Descending");
}

void selection()
{
    puts("MENU METODE SORTING");
    puts("1. Ascending");
    puts("2. Descending");
}

int main()
{
    struct timeval start, stop;
    int array[10] = {3, 10, 4, 6, 8, 9, 7, 2, 1, 5};
    int size = sizeof(array) / sizeof(array[0]);
    int pilihan, pilih;

    printArray(array, size);
    menu();
    printf("Pilihan Anda [1/2/3] : ");
    scanf("%d", &pilihan);
    switch (pilihan)
    {
    case 1:
        insertion();
        break;
    case 2:
        selection();
        break;
    case 3:
        exit(0);
    default:
        exit(0);
    }
    printf("Pilihan Anda [1/2] : ");
    scanf("%d", &pilih);
    gettimeofday(&start, NULL);
    if (pilihan == 1)
    {
```

```
        switch (pilih)
        {
        case 1:
            AscInsertionSort(array, size);
            break;
        case 2:
            DescInsertionSort(array, size);
            break;
        default:
            exit(0);
        }
    }
    else
    {
        switch (pilih)
        {
        case 1:
            AscSelectionSort(array, size);
            break;
        case 2:
            DescSelectionSort(array, size);
            break;
        default:
            exit(0);
        }
    }
    gettimeofday(&stop, NULL);
    printf("Perbandingan = %d\n", perbandingan);
    printf("Penukaran = %d\n", penukaran);
    printf("Sorting %lu us\n", ((stop.tv_sec - start.tv_sec) * 1000000 + stop.tv_usec -
start.tv_usec));
    printArray(array, size);
    return 0;
}
```

Output :

a. Insertion

- Ascending

```
3 10 4 6 8 9 7 2 1 5
MENU METODE SORTING
1. Insertion Sort
2. Selection Sort
3. Keluar
Pilihan Anda [1/2/3] : 1
MENU METODE SORTING
1. Ascending
2. Descending
Pilihan Anda [1/2] : 1
Perbandingan = 8
Penukaran = 8
Sorting 9 us
1 2 3 4 5 6 7 8 9 10
```

- Descending

```
3 10 4 6 8 9 7 2 1 5
MENU METODE SORTING
1. Insertion Sort
2. Selection Sort
3. Keluar
Pilihan Anda [1/2/3] : 1
MENU METODE SORTING
1. Ascending
2. Descending
Pilihan Anda [1/2] : 2
Perbandingan = 8
Penukaran = 7
Sorting 9 us
10 9 8 7 6 5 4 3 2 1
```

b. Selection

- Ascending

```
3 10 4 6 8 9 7 2 1 5
MENU METODE SORTING
1. Insertion Sort
2. Selection Sort
3. Keluar
Pilihan Anda [1/2/3] : 2
MENU METODE SORTING
1. Ascending
2. Descending
Pilihan Anda [1/2] : 1
Perbandingan = 45
Penukaran = 7
Sorting 13 us
1 2 3 4 5 6 7 8 9 10
```

- Descending

```
3 10 4 6 8 9 7 2 1 5
MENU METODE SORTING
1. Insertion Sort
2. Selection Sort
3. Keluar
Pilihan Anda [1/2/3] : 2
MENU METODE SORTING
1. Ascending
2. Descending
Pilihan Anda [1/2] : 2
Perbandingan = 45
Penukaran = 8
Sorting 12 us
10 9 8 7 6 5 4 3 2 1
```

4. 1000 data Array generate secara random. Kemudian diurutkan menggunakan Insertion dan juga Selection Sort

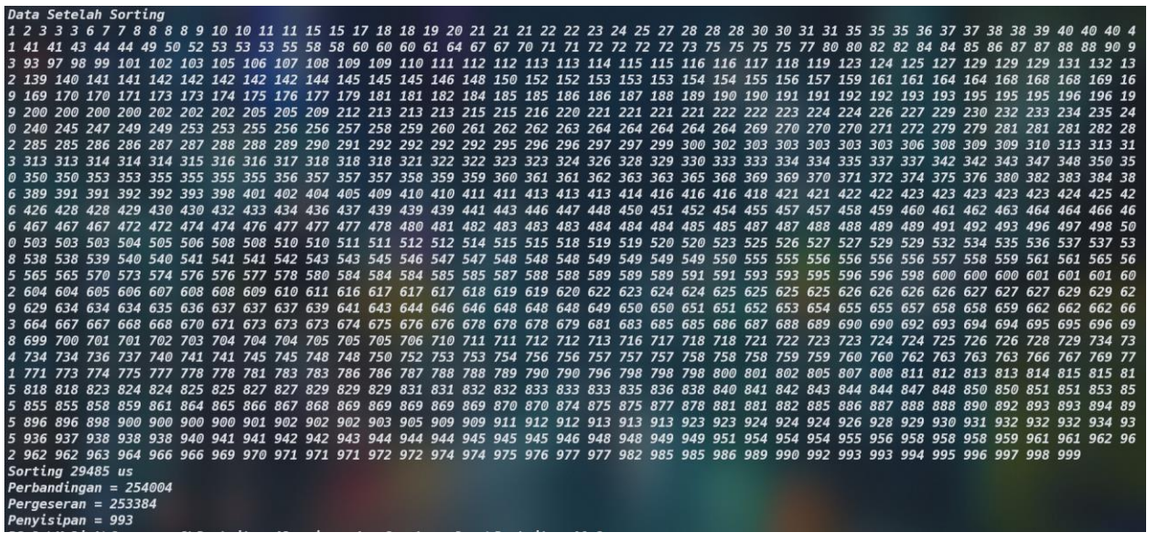
a. Insertion Sort

Kode :

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/time.h>
#include <unistd.h>
#define MAKS 1000

int perbandingan, pergeseran, penyisipan;

void printArray(int arr[], int sz)
{
   for (int i = 0; i < sz; i++)
   {
      printf("%d ", arr[i]);
   }
   printf("\n");
}

void insertionSort(int arr[], int sz)
{
   for (int step = 1; step < sz; step++)
   {
      int key = arr[step];
      int back = step - 1;

      // best case comparison is n-1

      while (key < arr[back] && back >= 0)
      {
         // worst case comparison is n(n-1)/2
         usleep(100);
         pergeseran++;
```

```
            arr[back + 1] = arr[back];
            back--;
            perbandingan++;
        }
        if (key > arr[back])
        {
            perbandingan++;
            usleep(100);
        }
        if (arr[back + 1] != key)
            penyisipan++;
        arr[back + 1] = key;
    }
}

int main()
{
    struct timeval start, stop;
    int array[MAKS];
    for (int i = 0; i < MAKS; i++)
    {
        array[i] = rand();
    }
    int size = sizeof(array) / sizeof(array[0]);

    puts("Data Sebelum Sorting");
    printArray(array, size);
    gettimeofday(&start, NULL);
    insertionSort(array, size);
    gettimeofday(&stop, NULL);
    puts("Data Setelah Sorting");
    printArray(array, size);
    printf("Sorting %lu us\n", ((stop.tv_sec - start.tv_sec) * 1000000 + stop.tv_usec -
start.tv_usec));
    printf("Perbandingan = %d\n", perbandingan);
    printf("Pergeseran = %d\n", pergeseran);
    printf("Penyisipan = %d\n", penyisipan);
    return 0;
}
```

Output :



b. Selection Sort

Kode :

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/time.h>
#include <unistd.h>
#define MAKS 1000

int perbandingan, penukaran;
void printArray(int arr[], int sz)
{
   for (int i = 0; i < sz; i++)
   {
      printf("%d ", arr[i]);
   }
   printf("\n");
}

void selectionSort(int arr[], int sz)
{
   for (int step = 0; step < sz - 1; step++)
   {
      int min = step;
      for (int i = step + 1; i < sz; i++)
      {
         usleep(100);
         perbandingan++;
         if (arr[i] < arr[min])
         {
            min = i;
         }
      }
      if (arr[min] != arr[step])
         penukaran++;
      int temp = arr[min];
      arr[min] = arr[step];
      arr[step] = temp;
   }
}

int main()
{
   struct timeval start, stop;
```

```
    int array[MAKS];
    for (int i = 0; i < MAKS; i++)
    {
       array[i] = rand() % 1000;
    }
    int size = sizeof(array) / sizeof(array[0]);

    puts("Data Sebelum Sorting");
    printArray(array, size);
    gettimeofday(&start, NULL);
    selectionSort(array, size);
    gettimeofday(&stop, NULL);
    puts("Data Setelah Sorting");
    printArray(array, size);
    printf("Sorting %lu us\n", ((stop.tv_sec - start.tv_sec) * 1000000 + stop.tv_usec -
start.tv_usec));
    printf("Perbandingan = %d\n", perbandingan);
    printf("Penukaran = %d\n", penukaran);
    return 0;
}
```

Output :

```
Data Sebelum Sorting
41 467 334 500 169 724 478 358 962 464 705 145 281 827 961 491 995 942 827 436 391 604 902 153 292 382 421 716 718 895 447 726 771 538 869 912 6
67 299 35 894 703 811 322 333 673 664 141 711 253 868 547 644 662 757 37 859 723 741 529 778 316 35 190 842 288 106 40 942 264 648 446 805 890 7
29 370 350 6 101 393 548 629 623 84 954 756 840 966 376 931 308 944 439 626 323 537 538 118 82 929 541 833 115 639 658 704 930 977 306 673 386 2
1 745 924 72 270 829 777 573 97 512 986 290 161 636 355 767 655 574 31 52 350 150 941 724 966 430 107 191 7 337 457 287 753 383 945 909 209 758
221 588 422 946 506 30 413 168 900 591 762 655 410 359 624 537 548 483 595 41 602 350 291 836 374 20 596 21 348 199 668 484 281 734 53 999 418 9
38 900 788 127 467 728 893 648 483 807 421 310 617 813 514 309 616 935 451 600 249 519 556 798 303 224 8 844 609 989 702 195 485 93 343 523 587
314 503 448 200 458 618 580 796 798 281 589 798 9 157 472 622 538 292 38 179 190 657 958 191 815 888 156 511 202 634 272 55 328 646 362 886 875
433 869 142 844 416 881 998 322 651 21 699 557 476 892 389 75 712 600 510 3 869 861 688 401 789 255 423 2 585 182 285 88 426 617 757 832 932 169
 154 721 189 976 329 368 692 425 555 434 549 441 512 145 60 718 753 139 423 279 996 687 529 549 437 866 949 193 195 297 416 286 105 488 282 455
734 114 701 316 671 786 263 313 355 185 53 912 808 832 945 313 756 321 558 646 982 481 144 196 222 129 161 535 450 173 466 44 659 292 439 253 24
 154 510 745 649 186 313 474 22 168 18 787 905 958 391 202 625 477 414 314 824 334 874 372 159 833 70 487 297 518 177 773 270 763 668 192 985 10
2 480 213 627 802 99 527 625 543 924 23 972 61 181 3 432 505 593 725 31 492 142 222 286 64 900 187 360 413 974 270 170 235 833 711 760 896 667 2
85 550 140 694 695 624 19 125 576 694 658 302 371 466 678 593 851 484 18 464 119 152 800 87 60 926 10 757 170 315 576 227 43 758 164 109 882 86
565 487 577 474 625 627 629 928 423 520 902 962 123 596 737 261 195 525 264 260 202 116 30 326 11 771 411 547 153 520 790 924 188 763 940 851 66
2 829 900 713 958 578 365 7 477 200 58 439 303 760 357 324 477 108 113 887 801 850 460 428 993 384 405 540 111 704 835 356 72 350 823 485 556 21
6 626 357 526 357 337 271 869 361 896 22 617 112 717 696 585 41 423 129 229 565 559 932 296 855 53 962 584 734 654 972 457 369 532 963 607 483 9
11 635 67 848 675 938 223 142 754 511 741 175 459 825 221 870 626 934 205 783 850 398 279 701 193 734 637 534 556 993 176 705 962 548 881 300 41
3 641 855 855 142 462 611 877 424 678 752 443 296 673 40 313 875 72 818 610 17 932 112 695 169 831 40 488 685 90 497 589 990 145 353 314 651 740
 44 258 335 759 192 605 264 181 503 829 775 608 292 997 549 556 561 627 467 541 129 240 813 174 601 77 215 683 213 992 824 601 392 759 670 428 2
7 84 75 786 498 970 287 847 604 503 221 663 706 363 10 171 489 240 164 542 619 913 591 704 818 232 750 205 975 539 303 422 98 247 584 648 971 86
4 913 75 545 712 546 678 769 262 519 985 289 944 865 540 245 508 318 870 601 323 132 472 152 87 570 763 901 103 423 527 600 969 15 565 28 543 34
7 88 943 637 409 463 49 681 588 342 608 60 221 758 954 888 146 690 949 843 430 620 748 67 536 783 35 226 185 38 853 629 224 748 923 359 257 766
944 955 318 726 411 25 355 1 549 496 584 515 964 342 75 913 142 196 948 72 426 606 173 429 404 705 626 812 375 93 565 36 736 141 814 994 256 652
 936 838 482 355 15 131 230 841 625 11 637 186 690 650 662 634 893 353 416 452 8 262 233 454 303 634 303 256 148 124 317 213 109 28 200 80 318 8
58 50 155 361 264 903 676 643 909 902 561 489 948 282 653 674 220 402 923 831 369 878 259 8 619 971 3 945 781 504 392 685 313 698 589 722 938 37
 410 461 234 508 961 959 493 515 269 937 869 58 700 971 264 117 215 555 815 330 39 212 288 82 954 85 710 484 774 380 815 951 541 115 679 110 898
 73 788 977 132 956 689 113 8 941 790 723 363 28 184 778 200 71 885 974 71 333 867 153 295 168 825 676 629 650 598 309 693 686 80 116 249
```

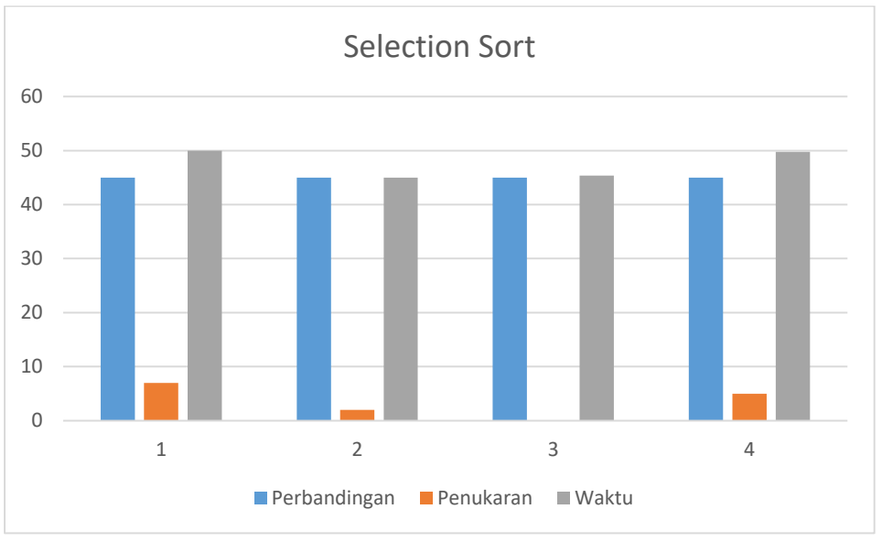
```
Data Setelah Sorting
1 2 3 3 3 6 7 7 8 8 8 8 9 10 10 11 11 15 15 17 18 18 19 20 21 21 21 22 22 23 24 25 27 28 28 28 30 30 31 31 35 35 35 36 37 37 38 38 39 40 40 40 4
1 41 41 43 44 44 49 50 52 53 53 53 55 58 58 60 60 60 61 64 67 67 70 71 71 72 72 72 72 73 75 75 75 75 77 80 80 82 82 84 84 85 86 87 87 88 88 90 9
3 93 97 98 99 101 102 103 105 106 107 108 109 109 110 111 112 112 113 113 114 115 115 116 116 117 118 119 123 124 125 127 129 129 129 131 132 13
2 139 140 141 141 142 142 142 142 144 145 145 145 146 148 150 152 152 153 153 153 154 154 155 156 157 159 161 161 164 164 168 168 168 169 16
9 169 170 170 171 173 173 174 175 176 177 179 181 181 182 184 185 185 186 186 187 188 189 190 190 191 191 192 192 193 193 195 195 195 196 196 19
9 200 200 200 200 202 202 202 205 205 209 212 213 213 213 215 215 216 220 221 221 221 221 222 222 223 224 224 226 227 229 230 232 233 234 235 24
0 240 245 247 249 249 253 253 255 256 256 257 258 259 260 261 262 262 263 264 264 264 264 264 269 270 270 270 271 272 279 279 281 281 281 282 28
2 285 285 286 286 287 287 288 288 289 290 291 292 292 292 292 295 296 296 297 297 299 300 302 303 303 303 303 303 306 308 309 309 310 313 313 31
3 313 313 314 314 314 315 316 316 317 318 318 318 321 322 322 323 323 324 326 328 329 330 333 333 334 334 335 337 337 342 342 343 347 348 350 35
0 350 350 353 353 355 355 355 355 356 357 357 357 358 359 359 360 361 361 362 363 363 365 368 369 369 370 371 372 374 375 376 380 382 383 384 38
6 389 391 391 392 392 393 398 401 402 404 405 409 410 410 411 411 413 413 413 414 416 416 416 418 421 421 422 422 423 423 423 423 424 425 42
6 426 428 428 429 430 430 432 433 434 436 437 439 439 439 441 443 446 447 448 450 451 452 454 455 457 457 458 459 460 461 462 463 464 464 466 46
6 467 467 467 472 472 474 474 476 477 477 477 478 480 481 482 483 483 483 484 484 484 485 485 487 487 488 488 489 489 491 492 493 496 497 498 50
0 503 503 503 504 505 506 508 508 510 510 511 511 512 512 514 515 515 518 519 519 520 520 523 525 526 527 527 529 529 532 534 535 536 537 537 53
8 538 538 539 540 540 541 541 541 542 543 543 545 546 547 547 548 548 548 549 549 549 549 550 555 555 556 556 556 556 557 558 559 561 561 565 56
5 565 565 570 573 574 576 576 577 578 580 584 584 584 585 585 587 588 588 589 589 589 591 591 593 593 595 596 596 598 600 600 600 601 601 601 60
2 604 604 605 606 607 608 608 609 610 611 616 617 617 617 618 619 619 620 622 623 624 624 625 625 625 625 626 626 626 626 627 627 627 629 629 62
9 629 634 634 634 635 636 637 637 637 639 641 643 644 646 646 648 648 648 649 650 650 651 651 652 653 654 655 655 657 658 658 659 662 662 662 66
3 664 667 667 668 668 670 671 673 673 673 674 675 676 676 678 678 678 679 681 683 685 685 686 687 688 689 690 690 692 693 694 694 695 695 696 69
8 699 700 701 701 702 703 704 704 704 705 705 705 706 710 711 711 711 712 712 713 716 717 717 718 718 721 722 723 723 724 724 725 726 726 728 729 734 73
4 734 734 736 737 740 741 741 745 745 748 748 750 752 753 753 754 756 756 757 757 757 758 758 758 759 759 760 760 762 763 763 763 766 767 769 77
1 771 773 774 775 777 778 778 781 783 783 786 786 787 788 788 789 790 790 796 798 798 798 800 801 802 805 807 808 811 812 813 813 814 815 815 81
5 818 818 823 824 824 825 825 827 827 829 829 829 831 831 832 832 833 833 835 836 838 840 841 842 843 844 844 847 848 850 850 851 851 853 85
5 855 855 858 859 861 864 865 866 867 868 869 869 869 869 869 870 870 874 875 875 877 878 881 881 882 885 886 887 888 888 890 892 893 893 894 89
5 896 896 898 900 900 900 900 901 902 902 902 903 905 909 909 911 912 912 913 913 913 923 923 924 924 924 926 928 929 930 931 932 932 932 934 93
5 936 937 938 938 938 940 941 941 942 942 943 944 944 944 945 945 945 946 948 948 949 949 951 954 954 954 955 956 958 958 958 959 961 961 962 96
2 962 962 963 964 966 966 969 970 971 971 971 972 972 974 974 975 976 977 977 982 985 985 986 989 990 992 993 993 994 995 996 997 998 999
Sorting 50458 us
Perbandingan = 499500
Penukaran = 995
```

Analisa :

1) Selection Sort

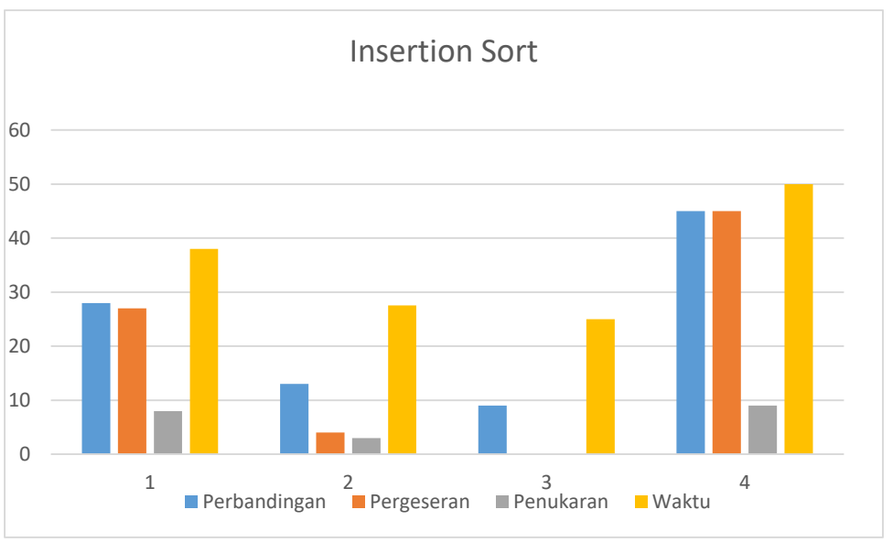
Perbandingan	Penukaran	Waktu
45	7	50
45	2	45

45	0	45.36610268
45	5	49.77902567



2) Insertion Sort

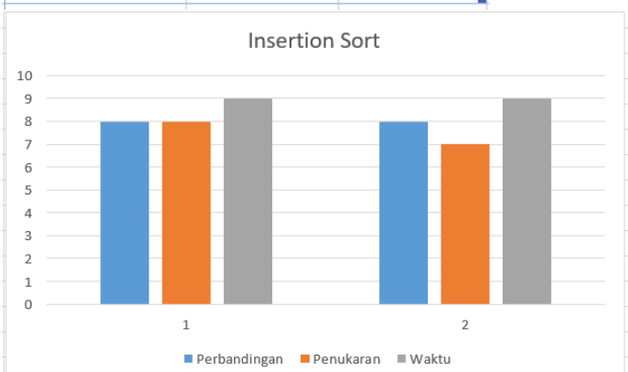
Perbandingan	Pergeseran	Penukaran	Waktu
28	27	8	37.99861659
13	4	3	27.5385121
9	0	0	25
45	45	9	50



Menu

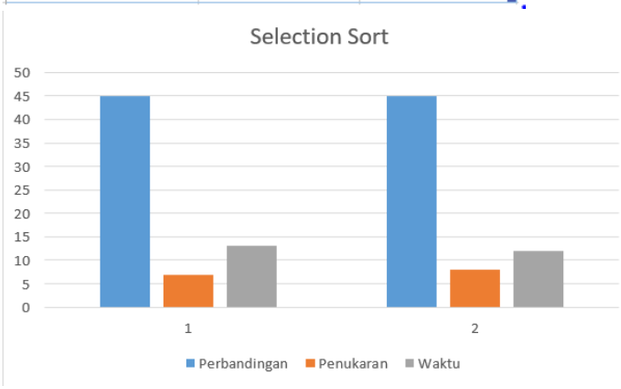
1) Insertion Sort

Perbandingan	Penukaran	waktu
8	8	9
8	7	9



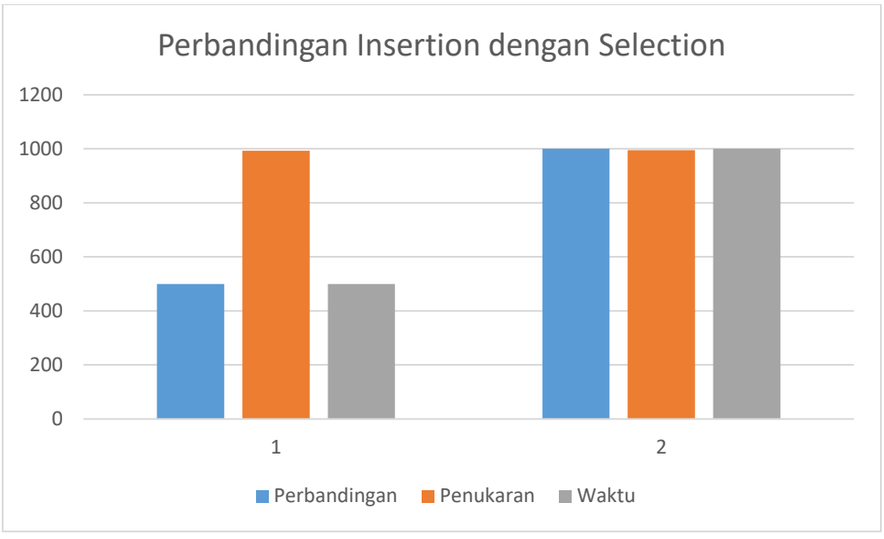
2) Selection Sort

Perbandingan	Penukaran	waktu
45	7	13
45	8	12



Generete nilai random

Perbandingan	Penukaran	Waktu
500	993	500
1000	995	1000



Kesimpulan

Waktu kompleksitas pada Selection sort pada semua kasus yaitu $O(n^2)$. Sedangkan Insertion sort antara best case dan worst casenya berbeda. Insertion sort best case memiliki waktu kompleksitas O(n) dan untuk worst casenya $O(n^2)$. Jadi apabila membicarakan mana yang lebih cepat antara Selection sort dan Insertion sort maka insertion akan lebih cepat karena lebih stabil. Semakin sedikit perbandingan yang dilakukan maka semakin cepat pula proses pengurutan. Beda halnya dengan selection sort. Apabila data banyak mau sudah terurut atau belum maka kecepatannya sama. Hal ini didasari oleh waktu komplesitas di atas.