# LAPORAN

**Algoritma dan Struktur Data**

A logo with a yellow and blue design

Description automatically generated

**Nama : Arif Muhammad Ihsan Marbun**

**Kelas : 1 D4 Teknik Informatika A**

**NRP : 3124600001**

1. Listing Latihan
2. Menggunakan merge and quick dengan struct

|  |
| --- |
| }  else  {  switch (urut){  case 1:  kondisi = data[j].no > data[maxmin].no;  break;  case 2:  kondisi = strcmpi (data[j].nama, data[maxmin].nama) > 0;  break;  case 3:  kondisi = data[j].nama > data[maxmin].nama;  break;  }  }  if (kondisi){  maxmin = j;  }  j++;  c++;  }  tukar (&data[maxmin], &data[i]);  s++;  }  }  void bubble (siswa data[]){  int batas = n-1;  for (i=0; i<n-1; i++){  did\_swap = 1;  if (did\_swap){  did\_swap = 0;  for (j=0; j<batas; j++){  if (jenis == 1){  switch (urut){  case 1:  kondisi = data[j].no > data[j+1].no;  break;  case 2:  kondisi = strcmpi (data[j].nama, data[j+1].nama) > 0;  break;  case 3:  kondisi = data[j].nama > data[j+1].nama;  break;  }  }  else {  switch (urut){  case 1:  kondisi = data[j].no < data[j+1].no;  break;  case 2:  kondisi = strcmpi (data[j].nama, data[j+1].nama) > 0;  break;  case 3:  kondisi = data[j].nama < data[j+1].nama;  break;  }  }  if (kondisi){  tukar(&data[j], &data[j+1]);  did\_swap = 1;  s++;  }  c++;  }  batas--;  }  }  }  void shell(siswa data[]){  int jarak = n/2;  while (jarak >= 1){  did\_swap = 1;  while (did\_swap){  did\_swap = 0;  for (i=0; i<n-jarak; i++){  if (jenis == 1){  switch (urut){  case 1:  kondisi = data[j].no < data[jarak+1].no;  break;  case 2:  kondisi = strcmpi (data[j].nama,data[jarak+1].nama) < 0;  break;  case 3:  kondisi = data[j].nama < data[jarak+1].nama;  break;  }  }  else {  switch (urut){  case 1:  kondisi = data[j].no > data[jarak+1].no;  break;  case 2:  kondisi = strcmpi (data[j].nama, data[jarak+1].nama) > 0;  break;  case 3:  kondisi = data[j].nama > data[jarak+1].nama;  break;  }  }  if (kondisi){  tukar(&data[i], &data[jarak+i]);  did\_swap = 1;  s++;  }  c++;  }  }  jarak /= 2;  }  }  void mergesort (siswa data[], int l, int r){  int med;  siswa hasil[n];  if (l < r){  med = (l+r) / 2;  mergesort(data, l, med);  mergesort(data, med+1, r);  merge(data, hasil, l, med, r);  }  }  void merge (siswa data[], siswa hasil[], int l, int med, int r){  int kiri1, kiri2, kanan1, kanan2;  kiri1 = l;  kiri2 = med+1;  kanan1 = med;  kanan2 = r;  i = l;  while (kiri1<=kanan1 && kiri2<=kanan2){  if (jenis == 1){  switch (urut){  case 1:  kondisi = data[kiri1].no <= data[kiri2].no;  break;  case 2:  kondisi = strcmpi(data[kiri1].nama,data[kiri2].nama) <= 0;  break;  case 3:  kondisi = data[kiri1].nilai <= data[kiri2].nilai;  break;  }  }  else {  switch (urut){  case 1:  kondisi = data[kiri1].no >= data[kiri2].no;  break;  case 2:  kondisi = strcmpi(data[kiri1].nama,data[kiri2].nama) >= 0;  break;  case 3:  kondisi = data[kiri1].nilai >= data[kiri2].nilai;  break;  }  }  if (kondisi){  hasil[i] = data[kiri1];  kiri1++;  c++;  m++;  }  else {  hasil[i] = data[kiri2];  kiri2++;  c++;  m++;  }  i++;  }  while (kiri1 <= kanan1){  hasil[i] = data[kiri1];  kiri1++;  i++;  m++;  }  while (kiri2 <= kanan2){  hasil[i] = data[kiri2];  kiri2++;  i++;  m++;  }  j = l;  while (j <= r){  data[j] = hasil[j];  j++;  }  }  void quick (siswa data[], int p, int r){  if (p < r){  q = partition (data, p, r);  quick (data, p, q);  quick (data, q+1, r);  }  }  int partition (siswa data[], int p, int r){  int step;  siswa pivot;  pivot = data[p];  i = p;  j = r;  while (i < j){  step = 1;  cek\_as\_des(data, pivot, step, j);  step = 2;  while (kondisi){  j--;  kondisi = cek\_as\_des(data, pivot, step, j);  c++;  }  step = 3;  kondisi2 = cek\_as\_des(data, pivot, step, i);  step = 4;  while (kondisi2){  i++;  kondisi2 = cek\_as\_des(data, pivot, step, i);  c++;  }  if (i < j){  tukar(&data[i], &data[j]);  s++;  i++;  j--;  }  }  return j;  }  int cek\_as\_des (siswa data[], siswa pivot, int step, int ij){  if ((jenis == 1 && step == 1) || (jenis != 1 && step == 3) || (jenis == 1 && step == 2) || (jenis != 1 && step == 4)){  switch (urut){  case 1:  kondisi = data[ij].no > pivot.no;  break;  case 2:  kondisi = strcmpi (data[ij].nama, pivot.nama) > 0;  break;  case 3:  kondisi = data[ij].nilai > pivot.nilai;  break;  }  }  else if ((jenis == 1 && step == 3) || (jenis != 1 && step == 1) || (jenis == 1 && step == 4)|| (jenis != 1 && step == 2)){  switch (urut){  case 1:  kondisi = data[ij].no < pivot.no;  break;  case 2:  kondisi = strcmpi (data[ij].nama, pivot.nama) < 0;  break;  case 3:  kondisi = data[ij].nilai < pivot.nilai;  break;  }  }  return (kondisi);  }  void tukar (siswa \*x, siswa \*y){  siswa temp;  temp = \*x;  \*x = \*y;  \*y = temp;  m += 3;  }  void tampil (siswa data[]){  puts("");  puts ("\tData yang terdaftar");  puts ("No\tNama\tNilai");  puts ("----------------------------------------");  for (i=0; i<n; i++){  printf ("%d\t%s\t%d\n", data[i].no, data[i].nama, data[i].nilai);  }  c\_m\_s ();  }  void c\_m\_s (){  printf ("Hasil compare : %d\n", c);  printf ("Hasil movement : %d\n", m);  printf ("Hasil swap : %d\n", s);  } |

A screen shot of a computer

AI-generated content may be incorrect. A screenshot of a computer

AI-generated content may be incorrect. A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.