**Nama : I Gede Gelgel Abdiutama**

**NIM : 2115101014**

**STRUKTUR DATA DAN ALGORITMA**

**UJIAN TENGAH SEMESTER**

**Deskripsi**

Pada ujian tengah semester ini terdapat 4 bagian soal yang harus anda kerjakan, diantaranya:

* **Bagian A :** Linked List
* **Bagian B :** Stack dan Queue
* **Bagian C :** Binary Search Tree
* **Bagian D :** Programming

Anda dapat mengerjakan ujian ini dengan menjawab pada lembar soal ini dan disimpan dalam format PDF. Jawaban dikumpulkan dengan format **SDA\_UTS\_[NIM].pdf**, sebagai contoh: SDA\_UTS\_1234567890.pdf. UTS dimulai pada hari Kamis, tanggal **14 April 2022, jam 14.00 WITA**. Batas waktu pengumpulan UTS adalah hari Kamis, tanggal **14 April 2022, jam 16.00 WITA**.

***Selamat bekerja!***

**BAGIAN A: LINKED LIST**

**Deskripsi**

Buatlah *pseudo-code* dan ilustrasi LinkedList sesuai dengan urutan perintah yang diberikan. *Pseudo-code* berupa langkah-langkah atau prosedur yang dilakukan pada saat program mengeksekusi perintah yang diberikan. Sedangkan ilustrasi LinkedList digambarkan dengan bagaimana bentuk LinkedList setelah perintah dieksekusi. Pada tugas ini, diasumsikan bahwa program memasukkan data ke LinkedList yang dibentuk secara **TERURUT DARI BESAR KE KECIL (Descending Ordered LinkedList)**.

**Contoh: 10 → 9→ 7 → 6 → 1**

**Soal A**

**Urutan Perintah:**

1. insert(138)
2. insert(283)
3. insert(220)
4. insert(197)
5. insert(174)
6. remove(138)
7. remove(283)

**Jawaban Soal A:**

|  |  |
| --- | --- |
| **Perintah** | **Prosedur Eksekusi** |
| insert(138) | tmp = new node();  tmp->value = 138;  head = tmp;   |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | head |  | 138 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  | null |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |
| insert(283) | tmp = new node();  tmp->value = 283;  tmp->next = head;  head = tmp;   |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | head |  | 283 |  | 138 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  | null |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |
| insert(220) | tmp = new node();  tmp->value = 220;  tmp->next = head->next;  head->next = tmp;   |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | head |  | 283 |  | 220 |  | 138 |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  | null |  |  |  |  |  |  |  |  |  |  |  |  | |
| insert(197) | tmp = new node();  tmp->value = 220;  tmp->next = head->next->next;  head->next = tmp;   |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | head |  | 283 |  | 220 |  | 197 |  | 138 |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  | null |  |  |  |  |  |  |  |  |  |  |  | |
| insert(174) | tmp = new node();  tmp->value = 174;  tmp->next = head->next->next->next;  head->next = tmp;   |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | head |  | 283 |  | 220 |  | 197 |  | 174 |  | 138 |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  |  |  | null |  |  |  |  |  |  |  |  |  |  | |
| remove(138) | tmp = head->next->next->next;  tmp2 = tmp->next;  tmp->next = tmp2->next;  delete(tmp2);   |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | head |  | 283 |  | 220 |  | 197 |  | 174 |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  |  | null |  |  |  |  |  |  |  |  |  |  | |
| remove(283) | tmp = head;  head = tmp->next;  delete(tmp);   |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | head |  |  | 220 |  | 197 |  | 174 |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  | null |  |  |  |  |  |  |  |  |  |  | |

**Daftar Perintah Berdasarkan NIM**

| NIM | Insert | | | | | Remove | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 2115101001 | 250 | 195 | 388 | 365 | 430 | 195 | 365 |
| 2115101002 | 482 | 475 | 300 | 205 | 198 | 300 | 198 |
| 2115101003 | 378 | 163 | 68 | 285 | 454 | 454 | 285 |
| 2115101004 | 274 | 91 | 452 | 229 | 70 | 70 | 163 |
| 2115101005 | 234 | 395 | 396 | 69 | 334 | 69 | 334 |
| 2115101006 | 314 | 387 | 244 | 405 | 222 | 314 | 405 |
| 2115101007 | 474 | 491 | 180 | 421 | 190 | 180 | 190 |
| 2115101008 | 114 | 435 | 108 | 253 | 478 | 435 | 253 |
| 2115101009 | 74 | 51 | 308 | 501 | 54 | 501 | 74 |
| 2115101010 | 498 | 203 | 468 | 173 | 86 | 498 | 173 |
| 2115101011 | 466 | 147 | 372 | 189 | 286 | 189 | 147 |
| 2115101012 | 426 | 419 | 212 | 293 | 270 | 270 | 293 |
| 2115101013 | 506 | 227 | 76 | 301 | 502 | 301 | 227 |
| 2115101014 | 138 | 283 | 220 | 197 | 174 | 138 | 283 |
| 2115101015 | 130 | 339 | 60 | 373 | 118 | 130 | 118 |
| 2115101016 | 202 | 123 | 196 | 125 | 374 | 202 | 123 |
| 2115101017 | 522 | 451 | 172 | 317 | 302 | 172 | 302 |
| 2115101018 | 258 | 411 | 92 | 349 | 438 | 92 | 349 |
| 2115101019 | 194 | 499 | 132 | 325 | 94 | 94 | 194 |
| 2115101020 | 282 | 427 | 492 | 461 | 318 | 318 | 282 |
| 2115101021 | 170 | 171 | 252 | 509 | 126 | 170 | 509 |
| 2115101022 | 178 | 523 | 52 | 477 | 390 | 390 | 178 |
| 2115101023 | 346 | 323 | 412 | 85 | 238 | 346 | 238 |
| 2115101024 | 122 | 355 | 324 | 357 | 166 | 166 | 355 |
| 2115101025 | 434 | 59 | 356 | 237 | 326 | 237 | 326 |
| 2115101026 | 242 | 67 | 332 | 109 | 102 | 242 | 102 |
| 2115101027 | 210 | 371 | 164 | 437 | 350 | 371 | 164 |
| 2115101028 | 298 | 363 | 228 | 149 | 358 | 363 | 228 |
| 2115101029 | 50 | 115 | 188 | 389 | 246 | 188 | 246 |
| 2115101030 | 322 | 507 | 292 | 53 | 182 | 292 | 322 |
| 2115101031 | 146 | 443 | 404 | 413 | 62 | 443 | 146 |
| 2115101032 | 442 | 267 | 460 | 245 | 398 | 460 | 245 |
| 2115101033 | 306 | 243 | 500 | 469 | 110 | 243 | 306 |
| 2115101034 | 154 | 179 | 340 | 309 | 278 | 278 | 179 |
| 2115101035 | 290 | 259 | 436 | 397 | 294 | 397 | 294 |
| 2115101036 | 330 | 315 | 268 | 333 | 310 | 310 | 315 |
| 2115101037 | 370 | 299 | 420 | 517 | 366 | 299 | 366 |
| 2115101038 | 98 | 483 | 444 | 93 | 446 | 93 | 446 |
| 2115101039 | 450 | 131 | 476 | 261 | 382 | 450 | 261 |
| 2115101040 | 362 | 459 | 364 | 141 | 518 | 362 | 364 |
| 2115101041 | 66 | 251 | 236 | 341 | 230 | 341 | 230 |
| 2115101042 | 58 | 99 | 316 | 213 | 526 | 526 | 316 |
| 2115101043 | 354 | 379 | 348 | 77 | 414 | 348 | 379 |
| 2115101044 | 458 | 291 | 284 | 445 | 214 | 458 | 214 |
| 2115101045 | 266 | 187 | 484 | 453 | 462 | 266 | 453 |
| 2115101046 | 90 | 139 | 260 | 101 | 78 | 260 | 101 |
| 2115101047 | 490 | 275 | 148 | 429 | 406 | 275 | 490 |
| 2115101048 | 386 | 155 | 100 | 61 | 158 | 386 | 155 |
| 2115101049 | 162 | 331 | 508 | 525 | 486 | 486 | 508 |
| 2115101050 | 226 | 219 | 84 | 493 | 206 | 206 | 219 |
| 2115101051 | 514 | 515 | 204 | 485 | 422 | 515 | 204 |
| 2115101052 | 218 | 107 | 380 | 277 | 510 | 277 | 510 |
| 2115101053 | 394 | 347 | 156 | 181 | 494 | 347 | 181 |
| 2115101054 | 402 | 211 | 516 | 133 | 142 | 516 | 211 |
| 2115101055 | 133 | 278 | 215 | 192 | 169 | 133 | 278 |
| 2115101056 | 125 | 334 | 55 | 368 | 113 | 125 | 113 |
| 2115101057 | 197 | 118 | 191 | 120 | 369 | 197 | 118 |
| 2115101058 | 517 | 446 | 167 | 312 | 297 | 167 | 297 |
| 2115101059 | 253 | 406 | 87 | 344 | 433 | 87 | 344 |
| 2115101060 | 189 | 494 | 127 | 320 | 89 | 89 | 189 |
| 2115101061 | 277 | 422 | 487 | 456 | 313 | 313 | 277 |
| 2115101062 | 165 | 166 | 247 | 504 | 121 | 165 | 504 |
| 2115101063 | 173 | 518 | 47 | 472 | 385 | 385 | 173 |
| 2115101064 | 341 | 318 | 407 | 80 | 233 | 341 | 233 |
| 2115101065 | 117 | 350 | 319 | 352 | 161 | 161 | 350 |
| 2115101066 | 429 | 54 | 351 | 232 | 321 | 232 | 321 |
| 2115101067 | 237 | 62 | 327 | 104 | 97 | 237 | 97 |
| 2115101068 | 205 | 366 | 159 | 432 | 345 | 366 | 159 |
| 2115101069 | 293 | 358 | 223 | 144 | 353 | 358 | 223 |
| 2115101070 | 45 | 110 | 183 | 384 | 241 | 183 | 241 |
| 2115101071 | 317 | 502 | 287 | 48 | 177 | 287 | 317 |
| 2115101072 | 141 | 438 | 399 | 408 | 57 | 438 | 141 |
| 2115101073 | 437 | 262 | 455 | 240 | 393 | 455 | 240 |
| 2115101074 | 301 | 238 | 495 | 464 | 105 | 238 | 301 |
| 2115101075 | 149 | 174 | 335 | 304 | 273 | 273 | 174 |
| 2115101076 | 285 | 254 | 431 | 392 | 289 | 392 | 289 |
| 2115101077 | 325 | 310 | 263 | 328 | 305 | 305 | 310 |
| 2115101078 | 365 | 294 | 415 | 512 | 361 | 294 | 361 |
| 2115101079 | 93 | 478 | 439 | 88 | 441 | 88 | 441 |
| 2115101080 | 445 | 126 | 471 | 256 | 377 | 445 | 256 |
| 2115101081 | 357 | 454 | 359 | 136 | 513 | 357 | 359 |

**BAGIAN B: Stack dan Queue**

**Deskripsi**

Terdapat 2 soal pada Bagian B di mana anda harus menggunakan Stack dan Queue untuk memecahkan suatu permasalahan struktur data. Berikut ini adalah contoh pengerjaan soal Bagian B:

**Contoh Soal:**

Terdapat sebuah stack (S1) dan sebuah queue (Q1) seperti ilustrasi berikut ini:

*Kondisi awal:*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| top | 10 |  |  |  |  |  | rear  front |
|  | 20 |  |  | 40 | 30 | 20 | 10 |
|  | 30 |  |  |  | **Q1** | |  |
|  | 40 |  |  |  |  |  |  |
|  | **S1** |  |  |  |  |  |  |

Tuliskan prosedur untuk memindahkan semua nilai dari S1 ke Q1 seperti ilustrasi berikut ini:

*Kondisi akhir:*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | rear |  |  | front |
|  |  |  |  | 40 | 30 | 20 | 10 |
|  |  |  |  |  | **Q1** | |  |
| top |  |  |  |  |  |  |  |
|  | **S1** |  |  |  |  |  |  |

**Contoh Jawaban:**

|  |  |
| --- | --- |
| **Prosedur** | **Ilustrasi** |
| tmp = S1.pop();  Q1.enqueue(tmp); | |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  | rear  front | | top | 20 |  |  | 40 | 30 | 20 | 10 | |  | 30 |  |  |  | **Q1** | |  | |  | 40 |  |  |  |  |  |  | |  | **S1** |  |  |  |  |  |  | |  |  |  |  |  |  |  |  | |
| tmp = S1.pop();  Q1.enqueue(tmp); | |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  | rear | front | |  |  |  |  | 40 | 30 | 20 | 10 | | top | 30 |  |  |  | **Q1** | |  | |  | 40 |  |  |  |  |  |  | |  | **S1** |  |  |  |  |  |  | |  |  |  |  |  |  |  |  | |
| tmp = S1.pop();  Q1.enqueue(tmp); | |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  | rear |  | front | |  |  |  |  | 40 | 30 | 20 | 10 | |  |  |  |  |  | **Q1** | |  | | top | 40 |  |  |  |  |  |  | |  | **S1** |  |  |  |  |  |  | |  |  |  |  |  |  |  |  | |
| tmp = S1.pop();  Q1.enqueue(tmp); | |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  | rear |  |  | front | |  |  |  |  | 40 | 30 | 20 | 10 | |  |  |  |  |  | **Q1** | |  | | top |  |  |  |  |  |  |  | |  | **S1** |  |  |  |  |  |  | |  |  |  |  |  |  |  |  | |

----Contoh soal dan contoh jawaban selesai----

**Soal B1**

Terdapat 2 buah Stack (S1 dan S2) dengan ilustrasi sebagai berikut:

*Kondisi awal:*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| top | 10 |  |  |  |
|  | 20 |  |  |  |
|  | 30 |  |  |  |
|  | 40 |  |  |  |
|  | 50 |  | top |  |
|  | **S1** |  |  | **S2** |

Tuliskan prosedur untuk memindahkan semua elemen yang ada di S1 ke S2 **dengan urutan yang sama** seperti ilustrasi berikut ini:

*Kondisi akhir:*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | top | 10 |  |  |  |  |  |
|  |  |  |  | 20 |  |  |  |  |  |
|  |  |  |  | 30 |  |  |  |  |  |
|  |  |  |  | 40 |  |  |  |  |  |
| top |  |  |  | 50 |  |  |  |  |  |
|  | **S1** |  |  | **S2** |  |  |  |  |  |

**Catatan:** anda dapat menambahkan Stack (S2, S3, …), Queue (Q2, Q3, …), atau variabel bantuan lainnya untuk menyelesaikan permasalahan di atas.

**Jawaban Soal B1:**

|  |  |
| --- | --- |
| **Prosedur** | **Ilustrasi** |
| tmp = S1.pop();  tmp = S3.pop();  tmp = S2.push(); | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  | 10 |  |  |  | 50 | | top | 20 |  |  |  | 40 | |  | 30 |  |  |  | 30 | |  | 40 |  |  |  | 20 | |  | 50 |  |  |  | 10 | |  | **S1** |  | **S2** |  | **S3** | |  |  |  |  |  |  | |
| tmp = S1.pop();  tmp = S3.pop();  tmp = S2.push(); | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  | 10 |  |  |  |  | | top | 20 |  |  |  | 40 | |  | 30 |  |  |  | 30 | |  | 40 |  |  |  | 20 | |  | 50 |  | 50 |  | 10 | |  | **S1** |  | **S2** |  | **S3** | |  |  |  |  |  |  | |
| tmp = S1.pop();  tmp = S3.pop();  tmp = S2.push(); | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  | 10 |  |  |  |  | | top | 20 |  |  |  |  | |  | 30 |  |  |  | 30 | |  | 40 |  | 40 |  | 20 | |  | 50 |  | 50 |  | 10 | |  | **S1** |  | **S2** |  | **S3** | |  |  |  |  |  |  | |
| tmp = S1.pop();  tmp = S3.pop();  tmp = S2.push(); | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  | 10 |  |  |  |  | | top | 20 |  |  |  |  | |  | 30 |  | 30 |  |  | |  | 40 |  | 40 |  | 20 | |  | 50 |  | 50 |  | 10 | |  | **S1** |  | **S2** |  | **S3** | |  |  |  |  |  |  | |
| tmp = S1.pop();  tmp = S3.pop();  tmp = S2.push(); | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  | 10 |  |  |  |  | | top | 20 |  | 20 |  |  | |  | 30 |  | 30 |  |  | |  | 40 |  | 40 |  |  | |  | 50 |  | 50 |  | 10 | |  | **S1** |  | **S2** |  | **S3** | |  |  |  |  |  |  | |
| tmp = S1.pop();  tmp = S3.pop();  tmp = S2.push(); | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  | 10 |  | 10 |  |  | | top | 20 |  | 20 |  |  | |  | 30 |  | 30 |  |  | |  | 40 |  | 40 |  |  | |  | 50 |  | 50 |  |  | |  | **S1** |  | **S2** |  | **S3** | |  |  |  |  |  |  | |

**Soal B2**

Terdapat dua buah Queue (Q1 dan Q2) seperti ilustrasi berikut ini:

*Kondisi awal:*

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | rear |  |  | front |  |  |  |  |  |  | rear  front |  |
|  | 10 | 20 | 30 | 40 |  |  |  |  |  |  |  |  |
|  |  | **Q1** | |  |  |  |  |  | **Q2** | |  |  |

Tuliskan prosedur untuk memindahkan seluruh elemen yang ada di Q1 ke Q2 **dengan urutan yang terbalik**, seperti pada ilustrasi berikut ini:

*Kondisi akhir:*

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | rear  front |  |  |  | rear |  |  | front |  |
|  |  |  |  |  |  |  |  | 40 | 30 | 20 | 10 |  |
|  |  | **Q1** | |  |  |  |  |  | **Q2** | |  |  |

**Catatan:** anda dapat menambahkan Stack (S2, S3, …), Queue (Q2, Q3, …), atau variabel bantuan lainnya untuk menyelesaikan permasalahan di atas.

**Jawaban Soal B2:**

|  |  |
| --- | --- |
| **Prosedur** | **Ilustrasi** |
| Q1.enqueue(tmp);  tmp = S1.pop();  Q2.enqueue(tmp); | |  |  |  |  |  | | --- | --- | --- | --- | --- | | rear | |  |  | front | | 10 | | 20 | 30 | 40 | |  | | **Q1** | |  | |  | |  | |  | |  | | 20 | | 30 | | 40 | | **S1** | |  | | rear | |  |  | front | |  | |  |  | 10 | |  | | **Q2** | |  | |  | | | | | | |
| Q1.enqueue(tmp);  tmp = S1.pop();  Q2.enqueue(tmp); | |  |  |  |  |  | | --- | --- | --- | --- | --- | | rear | |  |  | front | | 10 | | 20 | 30 | 40 | |  | | **Q1** | |  | |  | |  | |  | |  | |  | | 30 | | 40 | | **S1** | |  | | rear | |  |  | front | |  | |  | 20 | 10 | |  | | **Q2** | |  | |  | | | | | | |
| Q1.enqueue(tmp);  tmp = S1.pop();  Q2.enqueue(tmp); | |  |  |  |  |  | | --- | --- | --- | --- | --- | | rear | |  |  | front | | 10 | | 20 | 30 | 40 | |  | | **Q1** | |  | |  | |  | |  | |  | |  | |  | | 40 | | **S1** | |  | | rear | |  |  | front | |  | | 30 | 20 | 10 | |  | | **Q2** | |  | |  | | | | | | |
| Q1.enqueue(tmp);  tmp = S1.pop();  Q2.enqueue(tmp); | |  |  |  |  |  | | --- | --- | --- | --- | --- | | rear | |  |  | front | | 10 | | 20 | 30 | 40 | |  | | **Q1** | |  | |  | |  | |  | |  | |  | |  | |  | | **S1** | |  | | rear | |  |  | front | | 40 | | 30 | 20 | 10 | |  | | **Q2** | |  | |  | | | | | | |

**BAGIAN C: Binary Search Tree**

**Deskripsi**

Pada bagian C1 anda diminta untuk membentuk Binary Search Tree dengan daftar perintah yang diberikan. Pada C2 anda diminta untuk memberikan output Binary Tree Traversal berdasarkan algoritma yang diminta. Berikut ini adalah contoh untuk mengerjakan bagian C1 dan C2.

**Contoh Soal C1**

Bentuk struktur data Binary Tree berdasarkan perintah yang diberikan secara berurutan. Terdapat dua jenis perintah yaitu insert dan delete. Gambarkan **dua** binary tree yang terbentuk **setelah melakukan proses insert** dan **setelah melakukan proses delete**, seperti contoh di bawah ini:

**Urutan perintah:**

**Insert: 50, 30, 40**

**Delete: 40**

**Contoh Jawaban C1:**

|  |  |
| --- | --- |
| **Urutan Perintah** | **Binary Tree yang Terbentuk** |
| Insert: 50, 30, 40 |  |
| Delete: 40 |  |

**Contoh Soal C2.0:** Tuliskan output Pre-order Traversal berdasarkan output dari Contoh Jawaban C1.

**Contoh Jawaban C2.0:** 50, 30

**Soal C1**

Bentuk struktur data Binary Search Tree berdasarkan perintah yang diberikan secara berurutan sesuai dengan NIM masing-masing

**Daftar Perintah untuk Soal C1**

| NIM | Insert | | | | | | | | | | | | Delete | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| 2115101001 | 100 | 104 | 135 | 120 | 60 | 85 | 99 | 126 | 74 | 98 | 62 | 112 | 112 | 104 | 100 |
| 2115101002 | 100 | 67 | 122 | 86 | 107 | 104 | 69 | 90 | 105 | 94 | 119 | 73 | 119 | 122 | 100 |
| 2115101003 | 100 | 87 | 85 | 113 | 110 | 92 | 86 | 120 | 60 | 98 | 116 | 137 | 92 | 60 | 100 |
| 2115101004 | 100 | 65 | 62 | 80 | 69 | 109 | 85 | 136 | 90 | 127 | 115 | 135 | 115 | 136 | 100 |
| 2115101005 | 100 | 98 | 103 | 91 | 132 | 107 | 111 | 95 | 87 | 90 | 65 | 145 | 98 | 111 | 100 |
| 2115101006 | 100 | 116 | 93 | 85 | 99 | 94 | 115 | 70 | 135 | 123 | 112 | 95 | 85 | 112 | 100 |
| 2115101007 | 100 | 139 | 128 | 75 | 85 | 115 | 95 | 136 | 73 | 90 | 125 | 74 | 136 | 128 | 100 |
| 2115101008 | 100 | 130 | 114 | 102 | 80 | 70 | 85 | 111 | 141 | 65 | 75 | 95 | 141 | 130 | 100 |
| 2115101009 | 100 | 91 | 85 | 109 | 78 | 80 | 105 | 77 | 120 | 95 | 103 | 111 | 111 | 85 | 100 |
| 2115101010 | 100 | 76 | 80 | 96 | 69 | 125 | 115 | 110 | 82 | 98 | 105 | 120 | 120 | 110 | 100 |
| 2115101011 | 100 | 80 | 131 | 103 | 90 | 107 | 79 | 132 | 125 | 86 | 66 | 72 | 72 | 103 | 100 |
| 2115101012 | 100 | 93 | 117 | 98 | 112 | 121 | 105 | 67 | 130 | 89 | 86 | 66 | 130 | 89 | 100 |
| 2115101013 | 100 | 138 | 119 | 105 | 122 | 82 | 68 | 95 | 104 | 85 | 90 | 99 | 68 | 105 | 100 |
| 2115101014 | 100 | 72 | 90 | 116 | 134 | 89 | 105 | 80 | 95 | 112 | 92 | 131 | 80 | 134 | 100 |
| 2115101015 | 100 | 80 | 70 | 85 | 84 | 138 | 86 | 114 | 111 | 145 | 113 | 90 | 70 | 111 | 100 |
| 2115101016 | 100 | 117 | 134 | 110 | 80 | 115 | 70 | 86 | 94 | 98 | 125 | 73 | 98 | 134 | 100 |
| 2115101017 | 100 | 121 | 128 | 71 | 85 | 104 | 68 | 90 | 120 | 108 | 84 | 77 | 120 | 128 | 100 |
| 2115101018 | 100 | 130 | 73 | 119 | 93 | 95 | 89 | 132 | 133 | 110 | 68 | 87 | 132 | 87 | 100 |
| 2115101019 | 100 | 122 | 106 | 73 | 85 | 82 | 121 | 90 | 97 | 143 | 117 | 65 | 90 | 117 | 100 |
| 2115101020 | 100 | 68 | 95 | 84 | 130 | 104 | 60 | 85 | 90 | 117 | 137 | 134 | 90 | 137 | 100 |
| 2115101021 | 100 | 98 | 79 | 110 | 83 | 119 | 115 | 109 | 93 | 63 | 92 | 105 | 92 | 119 | 100 |
| 2115101022 | 100 | 122 | 71 | 85 | 70 | 89 | 83 | 107 | 120 | 66 | 128 | 137 | 70 | 137 | 100 |
| 2115101023 | 100 | 88 | 110 | 111 | 99 | 64 | 127 | 105 | 66 | 93 | 68 | 115 | 68 | 111 | 100 |
| 2115101024 | 100 | 65 | 85 | 70 | 93 | 110 | 75 | 120 | 107 | 123 | 60 | 108 | 75 | 120 | 100 |
| 2115101025 | 100 | 69 | 61 | 88 | 115 | 75 | 128 | 60 | 71 | 139 | 107 | 123 | 60 | 88 | 100 |
| 2115101026 | 100 | 95 | 113 | 72 | 110 | 74 | 127 | 79 | 78 | 61 | 112 | 124 | 78 | 95 | 100 |
| 2115101027 | 100 | 95 | 121 | 90 | 119 | 104 | 97 | 98 | 136 | 74 | 79 | 135 | 79 | 97 | 100 |
| 2115101028 | 100 | 118 | 113 | 115 | 80 | 94 | 93 | 63 | 131 | 85 | 84 | 125 | 93 | 84 | 100 |
| 2115101029 | 100 | 76 | 74 | 92 | 75 | 67 | 117 | 84 | 112 | 147 | 140 | 131 | 67 | 147 | 100 |
| 2115101030 | 100 | 80 | 86 | 126 | 71 | 69 | 124 | 138 | 85 | 125 | 129 | 90 | 71 | 129 | 100 |
| 2115101031 | 100 | 72 | 105 | 141 | 78 | 70 | 139 | 67 | 116 | 74 | 82 | 145 | 70 | 116 | 100 |
| 2115101032 | 100 | 102 | 125 | 131 | 72 | 82 | 70 | 81 | 85 | 122 | 65 | 110 | 102 | 110 | 100 |
| 2115101033 | 100 | 76 | 99 | 89 | 116 | 68 | 133 | 86 | 115 | 91 | 146 | 131 | 99 | 115 | 100 |
| 2115101034 | 100 | 108 | 95 | 106 | 134 | 86 | 113 | 65 | 68 | 85 | 90 | 107 | 85 | 95 | 100 |
| 2115101035 | 100 | 141 | 112 | 118 | 95 | 65 | 99 | 144 | 121 | 86 | 94 | 80 | 65 | 121 | 100 |
| 2115101036 | 100 | 90 | 80 | 115 | 131 | 117 | 70 | 124 | 85 | 86 | 107 | 95 | 95 | 90 | 100 |
| 2115101037 | 100 | 133 | 85 | 75 | 95 | 88 | 129 | 80 | 130 | 96 | 145 | 147 | 145 | 147 | 100 |
| 2115101038 | 100 | 137 | 81 | 86 | 76 | 106 | 105 | 84 | 117 | 89 | 109 | 95 | 89 | 105 | 100 |
| 2115101039 | 100 | 94 | 70 | 130 | 128 | 85 | 86 | 69 | 138 | 110 | 115 | 83 | 94 | 138 | 100 |
| 2115101040 | 100 | 65 | 82 | 138 | 106 | 88 | 70 | 122 | 104 | 92 | 115 | 89 | 88 | 89 | 100 |
| 2115101041 | 100 | 72 | 115 | 99 | 105 | 71 | 73 | 126 | 113 | 139 | 78 | 97 | 73 | 139 | 100 |
| 2115101042 | 100 | 127 | 129 | 108 | 64 | 95 | 126 | 72 | 71 | 90 | 93 | 110 | 64 | 110 | 100 |
| 2115101043 | 100 | 70 | 133 | 93 | 131 | 125 | 126 | 95 | 66 | 90 | 105 | 82 | 82 | 131 | 100 |
| 2115101044 | 100 | 123 | 86 | 114 | 97 | 121 | 120 | 125 | 77 | 90 | 95 | 85 | 95 | 114 | 100 |
| 2115101045 | 100 | 76 | 139 | 97 | 110 | 74 | 75 | 84 | 80 | 105 | 127 | 134 | 139 | 134 | 100 |
| 2115101046 | 100 | 85 | 95 | 75 | 103 | 80 | 70 | 134 | 132 | 122 | 106 | 94 | 132 | 106 | 100 |
| 2115101047 | 100 | 123 | 69 | 105 | 134 | 107 | 84 | 104 | 68 | 83 | 97 | 88 | 97 | 134 | 100 |
| 2115101048 | 100 | 113 | 80 | 115 | 77 | 110 | 103 | 104 | 78 | 85 | 84 | 75 | 110 | 75 | 100 |
| 2115101049 | 100 | 114 | 73 | 86 | 130 | 63 | 91 | 120 | 80 | 85 | 103 | 140 | 80 | 140 | 100 |
| 2115101050 | 100 | 110 | 94 | 85 | 119 | 105 | 64 | 115 | 90 | 95 | 87 | 123 | 90 | 64 | 100 |
| 2115101051 | 100 | 128 | 101 | 129 | 87 | 64 | 95 | 113 | 69 | 77 | 94 | 136 | 64 | 136 | 100 |
| 2115101052 | 100 | 88 | 70 | 134 | 119 | 135 | 68 | 75 | 82 | 118 | 106 | 98 | 82 | 119 | 100 |
| 2115101053 | 100 | 95 | 99 | 75 | 120 | 117 | 82 | 128 | 72 | 108 | 111 | 78 | 82 | 128 | 100 |
| 2115101054 | 100 | 131 | 88 | 70 | 108 | 123 | 103 | 80 | 121 | 69 | 76 | 85 | 88 | 121 | 100 |
| 2115101055 | 100 | 135 | 92 | 104 | 140 | 137 | 95 | 65 | 117 | 80 | 75 | 86 | 75 | 140 | 100 |
| 2115101056 | 100 | 71 | 116 | 85 | 72 | 88 | 68 | 106 | 94 | 130 | 133 | 125 | 68 | 88 | 100 |
| 2115101057 | 100 | 122 | 101 | 108 | 83 | 133 | 95 | 80 | 85 | 88 | 141 | 82 | 85 | 141 | 100 |
| 2115101058 | 100 | 121 | 99 | 70 | 85 | 65 | 71 | 125 | 118 | 137 | 122 | 72 | 72 | 99 | 100 |
| 2115101059 | 100 | 70 | 148 | 138 | 94 | 107 | 97 | 82 | 113 | 67 | 75 | 112 | 113 | 75 | 100 |
| 2115101060 | 100 | 144 | 99 | 103 | 102 | 105 | 75 | 84 | 95 | 87 | 63 | 126 | 99 | 126 | 100 |
| 2115101061 | 100 | 92 | 118 | 87 | 116 | 101 | 94 | 95 | 133 | 71 | 76 | 132 | 76 | 94 | 100 |
| 2115101062 | 100 | 115 | 110 | 112 | 77 | 91 | 90 | 60 | 128 | 82 | 81 | 122 | 90 | 81 | 100 |
| 2115101063 | 100 | 73 | 71 | 89 | 72 | 64 | 114 | 81 | 109 | 144 | 137 | 128 | 64 | 144 | 100 |
| 2115101064 | 100 | 77 | 83 | 123 | 68 | 66 | 121 | 135 | 82 | 122 | 126 | 87 | 68 | 126 | 100 |
| 2115101065 | 100 | 69 | 102 | 138 | 75 | 67 | 136 | 64 | 113 | 71 | 79 | 142 | 67 | 113 | 100 |
| 2115101066 | 100 | 99 | 122 | 128 | 69 | 79 | 67 | 78 | 82 | 119 | 62 | 107 | 99 | 107 | 100 |
| 2115101067 | 100 | 73 | 96 | 86 | 113 | 65 | 130 | 83 | 112 | 88 | 143 | 128 | 96 | 112 | 100 |
| 2115101068 | 100 | 105 | 92 | 103 | 131 | 83 | 110 | 62 | 65 | 82 | 87 | 104 | 82 | 92 | 100 |
| 2115101069 | 100 | 138 | 109 | 115 | 92 | 62 | 96 | 141 | 118 | 83 | 91 | 77 | 62 | 118 | 100 |
| 2115101070 | 100 | 87 | 77 | 112 | 128 | 114 | 67 | 121 | 82 | 83 | 104 | 92 | 92 | 87 | 100 |
| 2115101071 | 100 | 130 | 82 | 72 | 92 | 85 | 126 | 77 | 127 | 93 | 142 | 144 | 142 | 144 | 100 |
| 2115101072 | 100 | 134 | 78 | 83 | 73 | 103 | 102 | 81 | 114 | 86 | 106 | 92 | 86 | 102 | 100 |
| 2115101073 | 100 | 91 | 67 | 127 | 125 | 82 | 83 | 66 | 135 | 107 | 112 | 80 | 91 | 135 | 100 |
| 2115101074 | 100 | 62 | 79 | 135 | 103 | 85 | 67 | 119 | 101 | 89 | 112 | 86 | 85 | 86 | 100 |
| 2115101075 | 100 | 69 | 112 | 96 | 102 | 68 | 70 | 123 | 110 | 136 | 75 | 94 | 70 | 136 | 100 |
| 2115101076 | 100 | 124 | 126 | 105 | 61 | 92 | 123 | 69 | 68 | 87 | 90 | 107 | 61 | 107 | 100 |
| 2115101077 | 100 | 67 | 130 | 90 | 128 | 122 | 123 | 92 | 63 | 87 | 102 | 79 | 79 | 128 | 100 |
| 2115101078 | 100 | 120 | 83 | 111 | 94 | 118 | 117 | 122 | 74 | 87 | 92 | 82 | 92 | 111 | 100 |
| 2115101079 | 100 | 73 | 136 | 94 | 107 | 71 | 72 | 81 | 77 | 102 | 124 | 131 | 136 | 131 | 100 |
| 2115101080 | 100 | 82 | 92 | 72 | 100 | 77 | 67 | 131 | 129 | 119 | 103 | 91 | 129 | 103 | 100 |
| 2115101081 | 100 | 120 | 66 | 102 | 131 | 104 | 81 | 101 | 65 | 80 | 94 | 85 | 94 | 131 | 100 |

**Soal C1**

**Urutan perintah:**

**Insert: 100, 72, 90, 116, 134, 89, 105, 80, 95, 112, 92, 131**

**Delete: 80, 134, 100**

**Jawaban C1:**

|  |  |
| --- | --- |
| **Urutan Perintah** | **Binary Tree yang Terbentuk** |
| Insert: 100, 72, 90, 116, 134, 89, 105, 80, 95, 112, 92, 131 |  |
| Delete: 80, 134, 100 |  |

**Soal C2**

* **C2.1:** Tuliskan output Pre-Order Traversal berdasarkan output dari Jawaban C1!

Jawab : 105, 72, 90, 89, 95, 92, 116, 112, 131

* **C2.2:** Tuliskan output In-Order Traversal berdasarkan output dari Jawaban C1!

Jawab : 72, 89, 90, 92, 95, 105, 112, 116, 131

* **C2.3:** Tuliskan output Post-Order Traversal berdasarkan output dari Jawaban C1!

Jawab : 92, 95, 89, 90, 72, 112, 131, 116, 105

* **C2.4**: Gambarkan representasi binary tree di atas dengan array!

Jawab :

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | **20** | **21** |
| **Value** | **105** | **72** | **116** |  | **90** | **112** | **131** |  |  | **89** | **95** |  |  |  |  |  |  |  |  |  |  | **92** |



**BAGIAN D: Programming**

**Deskripsi**

Jawab pertanyaan-pertanyaan berikut ini:

**Soal D1.**

Perhatikan potongan kode berikut ini:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | // diketahui terdapat struktur data Binary Search Tree  void TraversalA(Node \*node){  if (node!=NULL){  TraversalA (node->rightChild);  cout << node->value;  TraversalA (node->leftChild);  }  }  void TraversalB(Node \*node){  if (node!=NULL){  if (node->left == NULL && node->right == NULL){  cout << node->value;  } else {  TraversalB (node->leftChild);  TraversalB (node->rightChild);  }  }  } |

Berdasarkan kode di atas, jawab pertanyaan berikut ini:

1. Algoritma apakah yang tertulis pada baris ke 3-9? Jelaskan cara kerjanya!

Jawab :

Algoritmanya yaitu In Order Traversal (Descending)

Cara Kerjanya yaitu mengurutkan data dengan urutan yang terbalik, ini dapat dibuktikan dengan sintax yang ada pada baris ke 5 dan 7, yang dimana algoritmanya akan mencari dan menampilkan paling kanan(value yang terbesar) terlebih dahulu, kemudian dilanjutkan kekiri(value yang terkecil).

1. Algoritma apakah yang tertulis pada baris ke 21-21? Jelaskan cara kerjanya!

Jawab :

Algoritmanya yaitu Mencari Leaf dan print semua Leaf yang ada pada Tree)

Cara Kerjanya yaitu mengeprint semua Leaf yang ada pada Tree yang dibuktikan dengan sintax pada baris ke 14, yang dimana menunjukkan sebuah kondisi jika Node Left dan Right nya adalah NULL maka akan di print value nya, namun ketika tidak ada maka akan dicari kekiri atau kekanan. Print Leaf itu akan dilakukan hanya jika Node yang dikir dan dikanan NULL/kosong.

**Soal D2**

Perhatikan potongan kode berikut ini:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26 | // diketahui dua buah pointer:  // Node \*front  // Node \*rear  void accodare(int el){  Node \*newEl = new Node(el);    if (rear==NULL){  front = newEl;  rear = newEl;  }else{  rear->next = newEl;  rear = newEl;  }  }  int togliereLaCoda(){  if (front==NULL)  return 0;    Node \*tmp = front;  front = front->next;    if (front==NULL)  rear = NULL;    return tmp->value;  } |

Berdasarkan kode di atas, jawab pertanyaan berikut ini:

1. Algoritma apakah yang tertulis pada baris ke 4-14? Jelaskan cara kerjanya!

Jawab :

Algoritmanya yaitu Enqueue

Cara Kerjanya yaitu jika dibagian front atau rear bernilai NULL, maka value nya yang dimasukkan akan menjadi front dan rear. Namun jika sudah terdapat value didalam front, maka value yang dimasukkan akan dijadikan elemen baru oleh rear.

1. Algoritma apakah yang tertulis pada baris ke 15-26? Jelaskan cara kerjanya!

Jawab :

Algoritmanya yaitu Dequeue

Cara Kerjanya yaitu ketika kondisi front bernilai NULL, maka akan di return. Namun jika Queue tidak NULL/berisi value maka front akan dikeluarkan dan value yang kedua sebelum front akan berubah menjadi front.