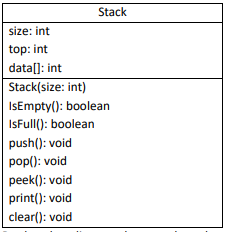
**Praktikum 1**

1. Perhatikan Diagram class berikut ini :



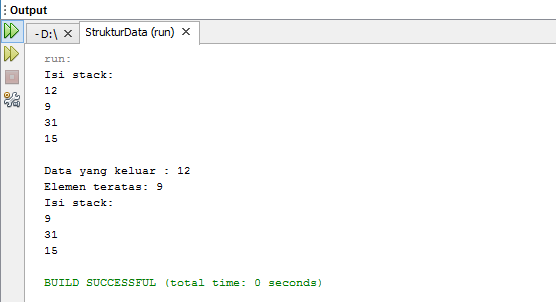
1. Pada Project StrukturData yang sudah dibuat pada Minggu sebelumnya, buat package dengan nama minggu8, kemudian buat class baru dengan nama Stack.

|  |  |
| --- | --- |
| No | Stack.java |
|  | package minggu8;  public class stack {  int size;  int top;  int data[];  public stack(int size) {  this.size = size;  data = new int[size];  top = -1;  }  public boolean IsEmpty(){  if (top == -1) {  return true;  }  else{  return false;  }  }    public boolean IsFull(){  if (top == size -1) {  return true;  }  else{  return false;  }  }    public void push(int dt){  if (!IsFull()) {  top++;  data[top] = dt;  }else{  System.out.println("Isi Stack Penuh");  }  }    public void pop(){  if (!IsEmpty()) {  int x = data[top];  top--;  System.out.println("Data yang keluar : " + x);  }else{  System.out.println("Stack masih kosong");  }  }    public void peek(){  System.out.println("Elemen teratas: " + data[top]);  }    public void print(){  System.out.println("Isi stack: ");  for (int i = top; i >= 0; i--) {  System.out.println(data[i]+ " ");  }  System.out.println(" ");  }    public void clear(){  if (!IsEmpty()) {  for (int i = top; i >=0; i--) {  top--;  }  System.out.println("Stack sudah dikosongkan");  }else{  System.out.println("Gagal! stack massih kosong");  }    }  } |

1. Buat class baru dengan nama StackMain tetap pada package minggu7. Buat class main, kemudian lakukan instansiasi objek dengan nama tumpukan dan nilai parameternya adalah 4.

|  |  |
| --- | --- |
| No | StackMain.java |
|  | public class StackMain {  public static void main(String[] args) {  stack tumpukan = new stack(4);  tumpukan.push(15);  tumpukan.push(31);  tumpukan.push(9);  tumpukan.push(12);    tumpukan.print();  tumpukan.pop();  tumpukan.peek();  tumpukan.print();  }  } |

1. Jalankan (run) class StackMain dan amati hasilnya



PERTANYAAN :

1. Perhatikan class StackMain, apakah fungsi angka 4 pada potongan kode program berikut?

stack tumpukan = new stack(4);

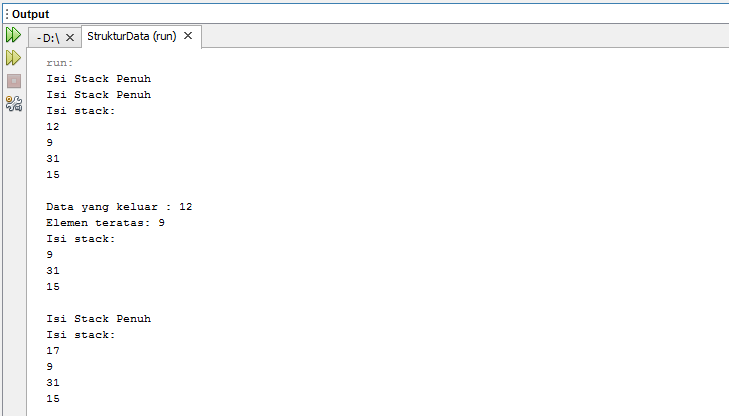
Jawab :

Untuk memasukkan angka sejumlah 4 kali atau 4 angka dimana ke empat angka tersebut di masukkan di dalam method push()

1. Lakukan penambahan data angka 17 dan 98 pada program. Tampilkan hasilnya!

Jawab :

|  |  |
| --- | --- |
| No | StackMain.java |
|  | public static void main(String[] args) {  stack tumpukan = new stack(4);  tumpukan.push(15);  tumpukan.push(31);  tumpukan.push(9);  tumpukan.push(12);  tumpukan.print();  tumpukan.pop();  tumpukan.peek();  tumpukan.print();    tumpukan.push(17);  tumpukan.push(98);  tumpukan.print();  } |



1. Pada soal nomor 2, mengapa data yang bisa dimasukkan ke dalam Stack hanya angka 17, sedangkan angka 98 tidak bisa? Jelaskan!

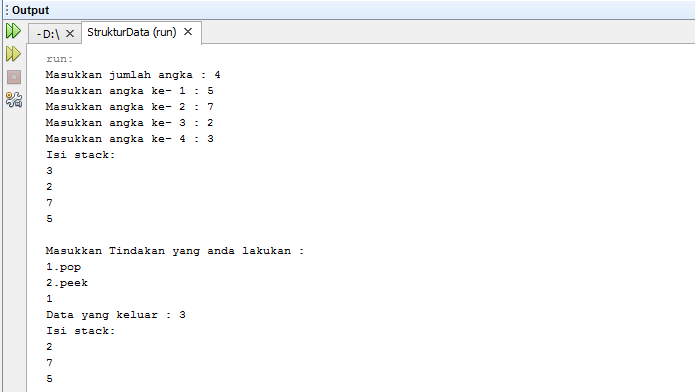
Jawab :

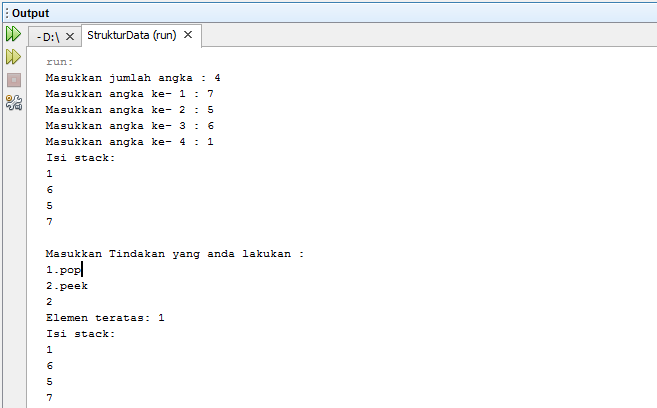
Data yang bisa diambah hanya 17 karena batas jumlah yang dimasukkan adalah 4. Hal ini berkaitan dengan instansiasi objek yang ada di soal 1. Peek(keluar) pertama adalah angka 12 sehingga jumlah data menjadi 3, kemudian push() kembali angka 17 sehingga jumlah data sekarang menjadi 4. Kemudian push kembali angka 98, namun output yang dihasilkan mengatakan bahwa “Isi Stack Penuh” karena jumlah data sudah 4 angka.

1. Modifikasi program tersebut sehingga pengguna dapat memasukkan data melalui keyboard (menggunakan Scanner)! Catatan: Anda dapat melakukan modifikasi dengan membentuk menu program

Jawab :

|  |  |
| --- | --- |
| No | StackMain.java |
|  | public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  System.out.print("Masukkan jumlah angka : ");  int bil = sc.nextInt();  stack tumpukan = new stack(bil);    for (int i = 1; i <=bil; i++) {  System.out.print("Masukkan angka ke- " + i + " : ");  int angka = sc.nextInt();  tumpukan.push(angka);  }    tumpukan.print();    System.out.println("Masukkan Tindakan yang anda lakukan : ");  System.out.println("1.pop");  System.out.println("2.peek");  int pilihan = sc.nextInt();  switch(pilihan){  case 1:  tumpukan.pop();  tumpukan.print();  break;  case 2:  tumpukan.peek();  tumpukan.print();  break;  default:  }  } |





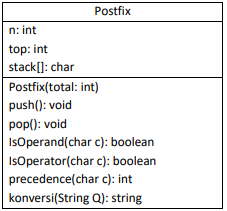
1. Pada konstruktor Stack, mengapa variabel top diinisialisasi dengan nilai -1?

Jawab :

Karena tipe data yang digunakan adalah tipe data array yang selalu dimulai dari indeks 0, maka top yang digunakan adalah -1.

**Praktikum 2**

1. Perhatikan Diagram Class berikut ini:



1. Pada package minggu8, buat class baru dengan nama Postfix. Tambahkan atribut n, top, dan stack sesuai dengan diagram class di atas.

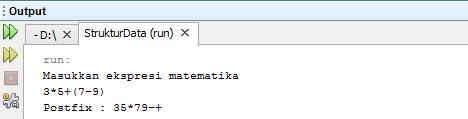
|  |  |
| --- | --- |
| No | Postfix.java |
|  | package minggu8;  import java.util.Scanner;  public class Postfix {  int n;  int top;  char stack[];    public Postfix(int total){  n = total;  top = -1;  stack = new char[n];  push('(');  }  public void push(char c){  top++;  stack[top] = c;  }    public char pop(){  char item = stack[top];  top--;  return item;  }    public boolean IsOperand(char c){  if ((c >= 'A' && c<= 'Z') || (c >= 'a' && c<='z') || (c >= '0' && c <= '9') || (c == ' ' && c == '.') ) {  return true;  }  else{  return false;  }  }    public boolean IsOperator(char c){  if (c== '^' || c == '%' || c == '/' || c == '\*' || c == '-' || c == '+') {  return true;  }  else{  return false;  }  }    public int presedence(char c){  switch(c){  case '^' : return 3;  case '%' : return 2;  case '/' : return 2;  case '\*' : return 2;  case '-' : return 1;  case '+' : return 1;  default: return 0;  }  }    public String koversi(String Q){  String P = "";  char c;  for (int i = 0; i < n; i++) {  c = Q.charAt(i);  if (IsOperand(c)) {  P = P +c;  }  if (c == '(') {  push(c);  }  if (c == ')') {  while (stack[top] != '(') {  P = P + pop();  }  char temp = pop();  }  if (IsOperator(c)) {  while (presedence(stack[top]) >= presedence(c)) {  P = P + pop();  }  push(c);  }  }  return P;  }  } |

1. Buat class baru dengan nama PostfixMain tetap pada package minggu7. Buat class main,

kemudian buat variabel P dan Q. Variabel P digunakan untuk menyimpan hasil akhir notasi postfix setelah dikonversi, sedangkan variabel Q digunakan untuk menyimpan masukan dari pengguna berupa ekspresi matematika dengan notasi infix. Buat scanner untuk menampung data masukan dari pengguna, kemudian panggil fungsi built-in trim yang digunakan untuk menghapus adanya spasi di depan atau di belakang teks

|  |  |
| --- | --- |
| No | PostfixMain.java |
|  | public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  String P,Q;  System.out.println("Masukkan ekspresi matematika");  Q = sc.next();  Q = Q.trim();  Q = Q + ')';    int total = Q.length();  Postfix post = new Postfix(total);  P = post.koversi(Q);  System.out.println("Postfix : " + P);    } |

1. Jalankan(run) class PostMain dan amati hasilnya



PERTANYAAN :

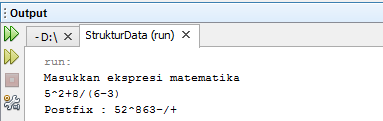
1. Perhatikan class Postfix,jelaskan fungsi dari method precedence!

Jawab :

Fungsi dari method precedence() adalah merupakan tingkat prioritas suatu algoritma aritmatika, dimana return 3 adalah prioritas utama yang harus dikerjakan terlebih dahulu sebelum 2 dan 1

1. Jalankan kembali program tersebut, masukkan ekspresi 5 ^ 2 + 8 / (6 – 3). Tampilkan hasilnya!

Jawab :

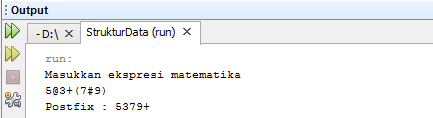


1. Pada soal nomor 2, mengapa tanda kurung tidak ditampilkan pada hasil konversi? Jelaskan!

Jawab :

1. Jalankan kembali program tersebut dengan memasukkan operator @ atau # pada persamaan matematika. Bagaimana hasilnya? Mengapa?

Jawab :



Program tetap dapat berjalan, namun keluaran yang dihasilkan adalah bahwa operator @ dan # diabaikan. Operator ini dianggap sebagai false sehingga tidak adakan terjadi apapun dari operator tersebut selain diabaikan. Hal ini bisa dilihat dari method IsOperator

