Sausan Irtiyaah (2408107010048)
Suci Wiranda (2408107010077)
UTS Struktur Data dan Algoritma

Konversi Aritmatika

Projek konversi aritmatika memuat bagaiman proses mengubah bentuk operasi bilangam menjadi ke bentuk lain namun tetap memiliki arti dan nilai yang sama. Konversi ini memuat infix, prefix, dan postfix. Infix memiliki simbol operasi di tengah, prefix di depan, dan postfix di belakang. Pada projek ini user membuat kode program berupa konversi aritmatika dengan beberapa fungsi yang nantinya akan dijalankan di menu utama melalui metode switch-case. Adapun fungsi yang digunakan adalah sebagai berikut:

1. Int space(char a)

   Fungsi ini digunakan untuk mengecek karakter yang dimasukkan spasi atau tab agar program tidak salah dalam membaca ekspresi perhitungan. Hal ini dikarenakan spasi atau tab tersebut tidak berguna dalan perhitungan, sehingga nantinya program akan mengabaikan spasi ataupun tab dalam proses kalkulasi.

2. Int precedence(char c)

   Fungsi ini digunakan untuk menentukan prioritas atau urutan eksekusi operator dalam ekspresi aritmatika, yang nantinya operator dengan prioritas lebih tinggi akan dieksekusi terlebih dahulu, contohnya "^ (pangkat)" akan lebih dulu dieksekusi dari pada "+ (tambah)".

3. Void push(char *c)

   Fungsi ini digunakan untuk menambahkan elemen ke dalam stack, fungsi ini nantinya akan menyimpan operator dan operand sementara dalam proses konversi.

4. Char pop()

   Fungsi ini digunakan untuk mengambil elemen terakhir dari stack atau mengambil operator dan operand yang sebelumnya disimpan sementara.

5. Int isEmpty()

   Fungsi ini digunakan untuk memeriksa apakah stack kosong atau tidak, yang mana dalam kode pada fungsi ini jika top = -1 maka stack kosong dan jika top != -1 maka stack berisi elemen.

6. Void reverseString(char *str)

   Fungsi ini digunakan untuk membalikkan string. Fungsi ini menggunakan dua indeks (i dan j) untuk menukar element pertama dan terakhir, elemen kedua dengan kedua terakhir, dan seterusnya. Fungsi ini digunakan dalam konversi infix ke prefix untuk membantu membaca dan memproses ekspresi dari kanan ke kiri.

7. Void pressEnterToContinue()

   Fungsi ini digunakan untuk menunggu pengguna menekan tombol “enter” sebelum eksekusi program di lanjutkan. Fungsi ini digunakan karena user memakai while sebagai loop untuk mengulang menu agar nantinya menu tidak otomatis muncul tanpa user minta.

8. Void intopost()

   Fungsi ini digunakan untuk mengubah ekspresi dari infix ke postfix. Dalam fungsi ini operand (angka/huruf) akan langsung di massukkan ke hasil, sedangkan operator akan di cek atau dibandingkan berdasarkan prioritasnya, sementara buka kurung “(“ disimpan dalam stack dan tutup kurung “)” mengeluarkan operator sampai bertemu dengan buka kurung “(“. Dan hasil akhirnya adalah ekspresi dari postfix.

9. Void posttoin()

   Fungsi ini digunakan untuk mengubah ekspresi postfix menjadi ekspresi infix. Dalam fungsi ini jika karakter adalah operand (angka/huruf) akan dimasukkan kedalam stack, jika karakter adalah operator (+, -, *, /, ^) maka program akan mengambil dua operator terakhir dari stack dan menggabungkannya menjadi ekspresi infix dengan tanda kurung dan dimasukkan lagi kedalam stack. Proses ini akan terus berulang sampai semua karakter di proses. Dan hasil akhirnya adalah ekspresi infix.

10. Void pretoin()

    Fungsi ini digunakan untuk mengubah ekspresi prefix menjadi ekspresi infix. Fungsi ini nantinya akan membaca ekspresi prefix dari kanan ke kiri, jika menemukan operand akan dimasukkan ke dalam stack, jika menemukan operator program akan mengambil dua operand terakhir dari stack dan menggabungkannya menjadi ekspresi infix lengkap dengan tanda kurung lalu memasukkannya kembali kedalam stack. Proses terus berulang sampai semua karakter diproses. Dengan demikian hasil akhirnya adalah ekspresi infix.

11. Void intopre(char *infix, char *prefix)

    Fungsi ini digunakan untuk mengubah ekspresi infix menjadi ekspresi prefix. Fungsi ini akan membalik ekspresi infix terlebih dahulu, setelahnya ekspresi yang telah dibalik akan

dibaca dari kiri ke kanan. Jika karakter addalah operand akan langsung di masukkan ke hasil, dan jika karakter adalah operator akan disimpan dalam stack dengan memperhatikan prioritasnya. Kemudian tutup kurung “)” dimasukkan kedalam stack dan buka kurung “(“ mengeluarkan operator sampai bertemu tutup kurung “)”. Setelah semua karakter di proses, operator yang ada di dalam stack akan dikeluarkan ke hasil. Karena awalnya infix dibalik, hasil yang diperoleh juga harus dibalik kembali untuk mendapatkan hasil akhirnya adalah ekspresi prefix.

12. Void pretopost(char *prefix, char *postfix)

    Fungsi ini digunakan untuk mengubah eksprefi prefix menjadi ekspresi postfix. Fungsi ini akan membaca ekspresi dari kanan ke kiri, jika karakter adalah operand akan dimasukkan kedalam stack, jika karakter adalah operator maka dua operan terakhir di dalam stack akan dikeluarkan dan digabung ke dalam bentuk postfix lalu hasilnya dimasukkan kembali ke dalam stack. Proses ini terus berulang sampai seluruh karakter di proses. Dengan demikian hasil akhirnya addalah ekspresi postfix.

13. Posttopre(char *postfix, char *prefix)

    Fungsi ini digunakan untuk mengubah ekspresi postfix menjadi ekspresi prefix. Fungsi ini akan membaca ekspresi dari kiri ke kanan, jika karakter adalah operand maka akan dimasukkan keddalam stack, jika karakter adalah operator maka akan diambil dua operan terkahir dari dalam stack lalu menggabungkannya dalam bentuk prefix kemudian memasukkannya kembali ke dalam stack. Proses terus berulang sampai semua karakter diproses dan mendapat hasil akhirnya berupa ekspresi prefix.

Semua fungsi di atas sudah memenuhi kebutuhan user untuk membuat menu yang diperlukan. Untuk menjalankan keseluruhan fungsi tersebut, maka user membuat fungsi utama yaitu **fungsi main** untuk menjalankan menu dan memanggil enam fungsi yang dibutuhkan (intopost, posttoin, pretoin, intopre, pretopost, dan posttopre). Fungsi ini akan menampilkan menu pilihan konversi. User memakai loop while yang bertujuan agar menu dapat diperiksa terlebih dahulu kondisi sebelum dieksekusi. Adapun user menggunakan switch-case untuk membuat menu konversi yang di enam case-nya memanggil satu fungsi konversi. Dengan demikian nanti pengguna akan diberikan menu pilihan untuk melakukan konversi ekspresi dan diminta untuk memilih opsi konversi yang diinginkan. Setelah memilih opsi, pengguna diminta untuk melakukan input berupa ekspresi yang sesuai dan selanjutnya program akan memanggil fungsi konversi yang

terkait. Adapula hal yang perlu diperhatikan adalah pengguna harus mengisi bentuk operasi dengan benar agar tidak terjadi kesalahan output. Kesalahan output dapat terjadi jika pengguna membuat operasi yang diminta dalam bentuk yang tidak valid.

Program ini menggunakan algoritma stack, karena stack merupakan struktur data yang sangat efisien dalam menangani pemrosesan dan konversi ekspresi aritmatika. Seperti yang kita tau stack bekerja dengan prinsip LIFO (Last In, First Out) yang memungkinkan data disimpan dan diambil dalam urutan yang sesuai dengan aturan prioritas operator dalam ekspresi matematika. Dalam program ini, nantinya stack digunakan untuk menyimpan operator dan operand sementara agar urutan eksekusi tetap sesuai aturan yang berlaku, terutama dalam konversi antara ekspresi infix, postfix, dan prefix.

Jumlah fungsi yang terdapat di dalam kode program (selain fungsi main)

Dalam program ini jumlah fungsi yang ada/digunakan adalah sebanyak 13 fungsi, yaitu :

1. Int space
2. Int precedence
3. Void push
4. Char *pop
5. Int isEmpty
6. Void reversestring
7. Void pressEnterToContiue
8. Void intopost
9. Void posttoin
10. Void pretoin
11. Void intopre
12. Void pretopost
13. Void posttopre