Programski prevodioci - predmetni zadatak

Osnovni podaci

Broj indeksa	Ime i prezime	Tema
SW1/2019	Jovan Jovančević	Strukture podataka niz i stek

Korišćeni alati

Naziv	Verzija
GCC compiler, Flex, Bison, Hipsim	

Evidencija implementiranog dela

Prenos niza po referenci – sintaksa i semantika
Niz kao povratna vrednost funkcija – sintaksa i semantika
Inicijalizacija niza pri deklaraciji – sintaksa, semantika i generisanje koda
Indeksiranje nizova – sintaksa, semantika i generisanje koda
Dodela vrednosti nizu na određenom indeksu – sintaksa, semantika i generisanje koda
Push operacija za stek - sintaksa, semantika i generisanje koda
Pop operacija za stek – sintaksa, semantika i generisanje koda
Top operacija za stek – sintaksa, semantika i generisanje koda

Detalji implementacije

Svi relevantni testovi nalaze se u korenskom direktorijumu projekta u folderima *stek* i *niz*.

- Implementirani su celobrojni nizovi koje je moguće (ali nije potrebno) inicijalizovati literalima u deklaraciji. Veličina niza je statička i ona se određuje pri deklaraciji. Elementima niza može se pristupiti korišćenjem literala, takođe na isti način se može raditi dodela vrednosti ovim nizovima. Vrednost koja se može dodeliti nizu može biti bilo koja numerička ekspresija.
- Takođe implementiran je i celobrojan stek, kojem se pri deklaraciji određuje veličina i ona je statička. Postavljanje nove operacije na stek je moguće uraditi operacijom *push*, i njoj se može proslediti bilo koja numerička ekspresija. Takođe skidanje elementa sa vrha steka moguće je odraditi operacijom *pop* nad stekom. Ukoliko je potrebno da se vrh steka samo pročita ali i ne izbaci sa steka to je moguće uraditi operacijom *top*.

Osnovna ideja za ove strukture podataka bila je sledeća:

- Ostavljanje mesta na steku (gde su elementi smeštani kada se radi generisanje koda)
- U tabeli simbola se čuva samo referenca (head) na niz/stek, var_num se povećava za onoliko koliko elemenata ima u određenoj strukturi podataka plus jedan. Da posle pri generisanju koda ne bi došlo do "gaženja" nekih informacija.
- Vrednost koju nosi *exp* ili *num_exp* više nije samo jedan integer, već struktura koja sadrži dva integer-a. Ovo proširenje bilo je potrebno zbog offset-a koje uvode nizovi i stek (pokazivač na vrh steka).
- U *gen_sym_name* funkciji dodata su još dva grananja za niz i za stek. Gde se generisanje „imena" radi slično kao i za obične varijable s tim da se doda još i offset pomnožen sa 4 koliko zauzima svaki element. Na slici 1 je prikazan kod funkcije.

```
// SYMBOL
void gen_sym_name(struct num_exp_vals *symbol)
{
    if (symbol->first > -1)
    {
        if (get_kind(symbol->first) == VAR) // -n*4(%14)
            code("-%d(%%14)", get_atr1(symbol->first) * 4);
        else if (get_kind(symbol->first) == STACK)
            code("-%d(%%14)", get_atr1(symbol->first) * 4 + (symbol->second + 1) * 4);
        else if (get_kind(symbol->first) == ARR) // -n*4(%14)
            code("-%d(%%14)", get_atr1(symbol->first) * 4 + (symbol->second + 1) * 4);
        else if (get_kind(symbol->first) == PAR) // m*4(%14)
            code("%d(%%14)", 4 + get_atr1(symbol->first) * 4);
        else if (get_kind(symbol->first) == LIT)
            code("$%s", get_name(symbol->first));
        else // function, reg
            code("%s", get_name(symbol->first));
    }
}
```

Slika 1

- Nakon promene ove funkcije nije bilo potrebno praviti velike izmene u *gen_mov* funkciji jer većinu posla obavlja prethodno navedena funkcija kao i sponuta struktura koja je nazvana *num_exp_vals*. Na slici 2 je prikazan kod funkcije.

```
void gen_mov(struct num_exp_vals *input_index, struct num_exp_vals *output_index)
{
    code("\n\t\tMOV \t");
    gen_sym_name(input_index);
    code(",");
    gen_sym_name(output_index);

    // ako se smešta u registar, treba preneti tip
    if (output_index->first >= 0 && output_index->first <= LAST_WORKING_REG)
        set_type(output_index->first, get_type(input_index->first));
    free_if_reg(input_index->first);
}
```

Slika 2

Niz

Primeri korišćenja:

- Inicijalizacija pri deklaraciji – *int niz = {1, 2 , 3};*
- Deklaracija niza određene dužine – *int niz[10];*
- Indeksiranje niza (koristimo niz kao izraz) – *var_a = niz[5];*
- Indeksiranje niza (iskaz) – *niz[1] = 11; niz[1] = a; niz[1] = niz2[1];*
- Element niza kao povratna vrednost – *return niz[1];*
- Niz kao povratna vrednost

- Potpis funkcije: *int* func() {...}*
- Return naredba: *return niz;*

Stek

Primeri korišćenja:

- Deklaracija steka određene dužine: *stack::int stek[5];*
- Push operacija: *stek.push(10); stek.push(a); stek.push(niz[1]);*
- Pop operacija*: stek.pop();*
- Top operacija*: stek.top();*
- Vrh steka kao povratna vrednost: *return stek.pop();*

Ideje za nastavak

- Pokušao sam, ali nisam uspeo da uradim indeksiranje nizova sa bilo kojim numeričkim izrazom. Razlog tome je to što se vrednost simbola ne čuva u tabeli simbola. S druge strane pri samom generisanje sym_name-a koje sam takođe pokušao da izmenim na takav način, da napravim pomoćni registar u pokušam u njega da stavim indeks te tako probam da uradim, nisam siguran da je sintaksno podržan (zato što vrednosti koje je hipsim generisao nisu imale smisla). Ukoliko bih to uspeo da implementiram, problem bi postala semantička provera, da li je index izvan opsega. Tako da je to jedna od ideja za nastavak.
- Dodatno, ideja za proširivanje projekta bilo bi generisanje koda za prenos niza po referenci.
- Kada bi se radilo o diplomskom radu, pokušao bih da implementiram agregatne funkcije za niz kao što su *Max, Min, Sum, Avg*...
- Takođe dodatan izazov bi bio da se omogući dinamičko proširivanje nizova.

Literatura

Materijali sa vežbi i predavanja
https://stackoverflow.com
https://www.programiz.com