Program Language homework2

jingshuai jiang jj2903

October 21, 2019

Exercise 1

a). $[a-z]^*[A-Z][a-z]^*[0-9][a-z]^*[A-Z][a-z]^*$

b).$([0-9]|[1-9][0-9]^*).[0-9][0-9]^*E([0-9]|[1-9][0-9]^*)$

c).$[A-Z](\epsilon|[A-Za-z0-9\_]|[A-Za-z0-9\_][A-Za-z0-9\_]|[A-Za-z0-9\_][A-Za-z0-9\_][A-Za-z0-9\_]|[A-Za-z0-9\_][A-Za-z0-9\_][A-Za-z0-9\_][A-Za-z0-9\_]|[A-Za-z0-9\_][A-Za-z0-9\_][A-Za-z0-9\_][A-Za-z0-9\_][A-Za-z0-9\_]|[A-Za-z0-9\_][A-Za-z0-9\_][A-Za-z0-9\_][A-Za-z0-9\_][A-Za-z0-9\_][A-Za-z0-9\_])$

Exercise 2

```
columns
    Prog => Parts
                |Parts Parts
    Parts => Dec
                |Fun
                |Proc
    Fun => fun name(P) Dec {content}
    Proc => proc name(P) Dec {content}
    content => content content
                  |E
                  |return name
    E=> EOE
            |name(V)
    O => =
            |-
            |*
            |/
            |+
    V => name
            |num
            |V,V
    P =>
            |P,P
            |Dec
    Dec => name:name
```

Exercise 3

a). *static scoping* : names used in a function are resolved in the environment of the function definition.

dynamic scoping : names used in a function are resolved in the environment of the function called.

b).

```
columns
    begin
    integer m, n;
    procedure A;
        begin
        print("in A : n = ", n);
        end;
    procedure B(n: integer);
        begin
        print("in B : m = ", m);
        print("in B : n = ", n);
        hardy;
        end;
    m := 50;
    n := 100;
    print("in main program : n = ", n);
    B(1);
    A;
    end;
```

The static scoping outputs:

in main program : n = 100

in B : m = 50

in B : n = 1

in A : n = 100

in A : n = 100

The dynamic scoping outputs:

in main program : n = 100

in B : m = 50

in B : n = 1

in A : n = 1

in A : n = 100

c). It will resolve the variable reference fisrt in the function itself. Then it will look for the variable from the function where it was defined.

d). It will resolve the variable reference fisrt in the function itself. Then it will look for the variable from the function where it was called.

Exercise 4

Exercise 5

a).pass by value: 2 4 6 8 10

b).pass by value: 2 11 6 8 10

c).pass by value: 2 7 6 8 10

d).pass by value: 2 4 6 8 11

Exercise 6

a).

```
columns
with text_io, ada.integer_text_io;
use text_io, ada.integer_text_io;
procedure mainprog is
package int_io is new integer_io (integer);
    use int_io;
    task oddtask is
        entry okdone;
    end oddtask;
    task eventask is
        entry okdone;
    end eventask;
    task body oddtask is
    begin
                accept okdone;
                for k in 1..100 loop
                ada.integer_text_io.put(k);
                if k rem 10 = 0 and k /=100 then
                eventask.okdone;
                accept okdone;
                end if;
                if k =100 then
                eventask.okdone;
                end if;
            end loop;
    end oddtask;
    task body eventask is
    begin
                accept okdone;
            for k in 201 .. 300 loop
              ada.integer_text_io.put(k);
              if k rem 10 = 0 and k /=300 then
              oddtask.okdone;
              accept okdone;
              end if;
            end loop;
    end eventask;
begin
        oddtask.okdone;
end mainprog;
```