PROGRAMMING METHODOLOGY

REPORT

INSIGHT

SUBMITTED TO:

PROGRAMMING METHODOLOGY LAB

SUBMITTED BY:

SAKSHI MAHESHWARI: 2016UCP1617

TWINKAL PARMAR : 2016UCP1700

Language Description

Insight version 1, deals with mathematical operation,logical operation,if else condition,loop function and so on.

Keywords :

1.**IF**:conditional statement

Syntax:[\nIF(logic exp to be checked)\nstatements~]~

2.**REST_ALL**:used in place of 'else'

Syntax:same as if.

It can only be used with 'if' condition.And it is optional.

3.**CURL**:used for loop value.

Syntax:[\nCURL(S@integer value:E@ integer value:g@integer value)

Action to be performed~]~

4.**S@**:used for denoting the starting of loop.

5.**E@**:used for denoting the ending of loop.

6. **G@**:used for denoting the gap of loop.

Operators:

1.**logical operator**:

a).<= b).>= c).== d).NOT= e).< f).> g).AND h).OR

2.**Airthmatic operator**:Used for calculating mathematical expression.

a).+ b).- c).* d)./ e).^

Functioning words:

Lexp:denotes logical expression
Mexp:denotes mathematical expression.

INBUILT FUNCTIONS:

1.**GRE_LE**:This function is used to identify the greatest and smallest number among the given values.
This can also used for characters values.and it will ans according to their ascii values.
Condition:only data of same datatypes should be include.
Syntax:GRE_LE(int,int,int,int……int)~

2.**ASC**:This function is very useful as it sorts the given values in ascending orders.Characters is sorted acc to their ascii values.
Condition : only one type of datatype can be used.
Syntax:ASC(int,int…….int)~

3.**DES**:This function is very useful as it sorts the given values in descending orders.Characters is sorted acc to their ascii values.
Condition : only one type of datatype can be used.
Syntax:DES(int,int…….int)~

How insight is different from other language and how it is useful?

Ans.

Insight is a basic programming language,have references from ‘C’ language.

The variable naming in insight is derived from c.

Insight syntax is completely different from C

Insight has certain in-built functions such as GRE-LE,ASC,DES.which helps the user or programmer to perform certain functions directly.

These functions can directly used to sort in orders as u want with just a function ,that reduces the complexity and do proper functioning.Helpful in many industrial work,can be used to store school and college data properly.

BNF

- ~$\n<stmt_queue>$~
- <stmt_queue> → <stmt>~"\n"<stmt_queue>
  |<stmt>~"\n"
- <stmt>→<mexp>|<lexp>|<if_st>|<loop_st>
- <mexp>→<id>=<term0>
- <term0>→<term0>+<term1>|
  <term0>-<term1>|<term1>
- <term1>→<term1>*<term2>|<term1>/<term2>|
  <term2>
- <term2>→<term3>^<term2>|<term3>
- <term3>→<fac> | (<mexp>)
- <lexp>→<lexp>OR<logic_exp1>|<logic_exp1>
- <logic_exp1>→<logic_exp1>AND<logic_exp2>|
  <logic_exp2>
- <logic_exp2>→NOT<exp>|<exp>
- <exp>→<fac><lopt><fac>|TRUE|FALSE
- <if_st>→"[\n"IF(<lexp>)"\n"<stmt_queue>
  [\nREST_ALL"\n"<stmt_queue>] ‘’]’’
- <loop_st>→’’[\n‘’CURL(S@<int>:E@<int>:G@<int>)’’\n’’
  <stmt_queue>’’]’’
- <lopt>=‘’==’’|’’<=’’|’’>=’’|’’<’’|’’>’’
- <fac>→<id>|<const>
- <const>→<int>
- <id>→ <alpha>|<alpha><id>|<id><int>
- <alpha>→a|b|c----x|y|z|A|B|C----X|Y|X|"_"

- <digit>→0|1|2|3|4|5|6|7|8|9

Regular expression

For <id>:

Gamma(Gamma|Beta)*

For <int>:

Beta(Beta)*

Gamma={a,b,c,……,z,A,B,C,…….Z,_}

Beta={0,1,2…….,9}

VALIDITY

1.~$\n

[

IF(a==5)

b=3~

]

~~

Derivation:

- ~$\n<stmt_queue>~~
- ~$\n <stmt>~\n ~~
- ~$\n<if_st>~\n ~~
- ~~ [\nIF(<lexp>)\n<stmt_queue>]~\n $~
- ~$\n[\nIF(<logic_exp1>)\n<stmt_queue>]~\n$~

- ~$\n[\nIF(<logic_exp2>)\n<stmt_queue>]~\n $~
- ~$\n[\nIF(<exp>))\n<stmt_queue>]~\n $~
- ~$\n[\nIF(<fac><lopt><fac>)\n<stmt_queue>]~\n $~
- * ~$\n[\nIF(<id>==<const>)\n<stmt_queue>]~\n $~
- * ~$\n[\nIF(<id>==<int>)\n<stmt>~\n]~\n$~
- * ~$\n[\nIF(<id>==<int>)\n<mexp>~\n]~\n$~
- ~$\n[\nIF(<id>==<int>)\n<id>=<term0>~\n]~\n$~
- ~$\n[\nIF(<id>==<int>)\n<id>=<term1>~\n]~\n$~
- ~$\n[\nIF(<id>==<int>)\n<id>=<term2>~\n]~\n$~
- ~$\n[\nIF(<id>==<int>)\n<id>=<term3>~\n]~\n$~
- ~$\n[\nIF(<id>==<int>)\n<id>=<fac>~\n]~\n$~
- ~$\n[\nIF(<id>==<int>)\n<id>=<const>~\n]~\n$~
- ~$\n[\nIF(<id>==<int>)\n<id>=<int>~\n]~\n$~
- * ~$\n[\nIF(a==5)\n b=3~\n]~\n$~

Hence it is valid:

INVALID

~~

a=5~

[

CURL(S@0:E@10:G@++1)

a=a+1~

]~

~~

DERIVATION

- ~$\n<stmt_queue> ~~
- ~$\n<stmt>~\n<stmt_queue>~~
- * ~$\n<mexp>~\n<stmt>~\n ~~
- * ~$\n<id>=<term0>~\n<loop_st>~\n$~
- *~$\n<id>=<term1>~\n[\nCURL(S@<t>:E@<t>:G@<int>)\n <stmt_queue>]~\n$~
- * ~$\n<id>=<term1>~\n[\nCURL(S@<t>:E@<t>:G@<int>)\n <stmt>]~\n$~

It is invalid in ‘’iNSIGHT’’ to write ‘G@++1’.As it only include int value here (acc to BNF).Correct syntax is G@1

Example set

~$

a=0~

[

CURL(S@0:E@11:G@2)

[

IF(a==4)

a=a+4~

REST_ALL

a=a+1~

]~

]~

~~

Derivation

- ~$\n<stme_queue>~~
- ~$\n<stmt>~\n<stmt_queue>~~
- * ~$\n<mexp>~\n<stmt>~\n$~
- * ~$\n<id>=<term0>~\n<loop_st>~\n$~
- * ~$\n<id>=<term1>~\n[\nCURL(S@<int>:E@<int>:G@<int>)
  \n<stmt_queue>]~\n$~
- * ~$\n<id>=<term2>~\n[\nCURL(S@<int>:E@<int>:G@<int>)\n
  <stmt>~\n]~\n$~
- * ~$\n<id>=<term3>~\n[\nCURL(S@<int>:E@<int>:G@<int>)\n
  <if_st>]~\n$~
- * ~$\n<id>=<fac>~\n[\nCURL(S@<int>:E@<int>:G@<int>)\n
  [\nIF(lexp)\n<stmt_queue>REST_ALL\n<stmt_queue>]~\n]~\n$~
- * ~$\n<id>=<int>~\n[\nCURL(S@<int>:E@<int>:G@<int>)\n
  [\nIF(logic_exp1)\n<stmt>~\nREST_ALL\n<stmt>~\n]~\n]~\n$~
- * ~$\n<id>=<int>~\n[\nCURL(S@<int>:E@<int>:G@<int>)\n
  [\nIF(logic_exp2)\n<mexp>~\nREST_ALL\n<mexp>~\n]~\n]~\n$~
- * ~$\n<id>=<int>~\n[\nCURL(S@<int>:E@<int>:G@<int>)\n
  [\nIF(lexp)\n<id>=<term0>~\nREST_ALL\n<id>=<term0>~\n]~\n]
  ~\n$~

- * ~$\n<id>=<int>~\n[\nCURL(S@<int>:E@<int>:G@<int>)\n [\nIF(logic_exp1)\n<id>=<term0>+<term1>~\nREST_ALL\n<id>= <term0>+<term1>~\n]~\n]~\n$~
- * ~$\n<id>=<int>~\n[\nCURL(S@<int>:E@<int>:G@<int>)\n [\nIF(logic_exp2)\n<id>=<term1>+<term2>~\nREST_ALL\n <id>= <term1>+<term2>~\n]~\n]~\n$~
- * ~$\n<id>=<int>~\n[\nCURL(S@<int>:E@<int>:G@<int>)\n [\nIF(exp)\n<id>=<term2>+<term3>~\nREST_ALL\n <id>= <term2>+<term3>~\n]~\n]~\n$~
- * ~$\n<id>=<int>~\n[\nCURL(S@<int>:E@<int>:G@<int>)\n [\nIF(<fac><lopt><fac>)\n<id>=<term3>+<fac>~\nREST_ALL\n <id>= <term3>+<fac>~\n]~\n]~\n$~
- * ~$\n<id>=<int>~\n[\nCURL(S@<int>:E@<int>:G@<int>)\n [\nIF(<id><lopt><const>)\n<id>=<fac>+<const>~\nREST_ALL\n <id>= <fac>+<const>~\n]~\n]~\n$~
- * ~$\n<id>=<int>~\n[\nCURL(S@<int>:E@<int>:G@<int>)\n [\nIF(<id><lopt><int>)\n<id>=<id>+<int>~\nREST_ALL\n <id>= <id>+<int>~\n]~\n]~\n$~
- * ~$\n a=0~\n[\nCURL(S@0:E@11:G@2)\n [\nIF(a==4)\n a=a+4~\nREST_ALL\n a= a+<const>~\n]~\n]~\n$~