Topic of Assignment:

"Slicing Program"

Student Name:

M. Taimoor
(FA22-BSE-072)

Subject:

Software Re-Engineering

Teacher Name:

Dr.Manzoor Ahmad

Program no. 1

Solution:

n	Statement	REFs(n)	DEFs(n)	relevant(n)
1	b=1		b	
2	c=2		c	{b}
3	d=3		d	{b}
4	a=d	d	a	{b, d}
5	if (a=3) then	a		{b, d}
6	d=b+d	b, d	d	{b, d}
7	c=b+d	b, d	c	{b, d}
8	else			{b, c}
9	b=b+1	b	b	{b, c}
10	d=b+1	b	d	{b, c}
11	Endif			{b, c}
12	a=b+c	b, c	a	{b, c}
13	print a	a		{a}

Program Slice on <13, a>:

{12, 9, 2, 1}

n	Statement
1	b=1
2	c=2

9	b=b+1
12	a=b+c

Program #1:
Detailed Calculation of relevant(n) for Slice <13, a>
Initialization:

- **relevant(13) = {a} (from the slicing criterion <13, a>)**

Step 1: Calculate relevant(12)

Statement 12: a = b + c
DEF(12) = {a}
REF(12) = {b, c}

relevant(12) = (relevant(13) - DEF(12)) ∪ (REF(12) if relevant(13) ∩ DEF(12) ≠ ∅)
= ({a} - {a}) ∪ ({b, c} if {a} ∩ {a} ≠ ∅)
= ∅ ∪ {b, c}
= {b, c}

Step 2: Calculate relevant(11)

Statement 11: Endif
DEF(11) = ∅
REF(11) = ∅

relevant(11) = (relevant(12) - DEF(11)) ∪ (REF(11) if relevant(12) ∩ DEF(11) ≠ ∅)
= ({b, c} - ∅) ∪ (∅ if {b, c} ∩ ∅ ≠ ∅)
= {b, c} ∪ ∅
= {b, c}

Step 3: Calculate relevant(10)

Statement 10: d = b + 1
DEF(10) = {d}
REF(10) = {b}

relevant(10) = (relevant(11) - DEF(10)) ∪ (REF(10) if relevant(11) ∩ DEF(10) ≠ ∅)
= ({b, c} - {d}) ∪ ({b} if {b, c} ∩ {d} ≠ ∅)
= {b, c} ∪ ∅
= {b, c}

Step 4: Calculate relevant(9)

Statement 9: b = b + 1
DEF(9) = {b}
REF(9) = {b}

relevant(9) = (relevant(10) - DEF(9)) ∪ (REF(9) if relevant(10) ∩ DEF(9) ≠ ∅)
= ({b, c} - {b}) ∪ ({b} if {b, c} ∩ {b} ≠ ∅)
= {c} ∪ {b}
= {b, c}

Step 5: Calculate relevant(8)

Statement 8: else
DEF(8) = ∅
REF(8) = ∅

relevant(8) = (relevant(9) - DEF(8)) ∪ (REF(8) if relevant(9) ∩ DEF(8) ≠ ∅)
= ({b, c} - ∅) ∪ (∅ if {b, c} ∩ ∅ ≠ ∅)
= {b, c}

Step 6: Calculate relevant(7)

Statement 7: c = b + d
DEF(7) = {c}
REF(7) = {b, d}

relevant(7) = (relevant(11) - DEF(7)) ∪ (REF(7) if relevant(11) ∩ DEF(7) ≠ ∅)
= ({b, c} - {c}) ∪ ({b, d} if {b, c} ∩ {c} ≠ ∅)
= {b} ∪ {b, d}
= {b, d}

Step 7: Calculate relevant(6)

Statement 6: d = b + d
DEF(6) = {d}
REF(6) = {b, d}

relevant(6) = (relevant(7) - DEF(6)) ∪ (REF(6) if relevant(7) ∩ DEF(6) ≠ ∅)
= ({b, d} - {d}) ∪ ({b, d} if {b, d} ∩ {d} ≠ ∅)
= {b} ∪ {b, d}
= {b, d}

Step 8: Calculate relevant(5)

Statement 5: if (a=3) then
DEF(5) = ∅
REF(5) = {a}

relevant(5) = (relevant(6) - DEF(5)) ∪ (REF(5) if relevant(6) ∩ DEF(5) ≠ ∅)
= ({b, d} - ∅) ∪ ({a} if {b, d} ∩ ∅ ≠ ∅)
= {b, d}

Step 9: Calculate relevant(4)

Statement 4: a = d
DEF(4) = {a}
REF(4) = {d}

relevant(4) = (relevant(5) - DEF(4)) ∪ (REF(4) if relevant(5) ∩ DEF(4) ≠ ∅)
= ({b, d} - {a}) ∪ ({d} if {b, d} ∩ {a} ≠ ∅)
= {b, d}

Step 10: Calculate relevant(3)

Statement 3: d = 3
DEF(3) = {d}
REF(3) = ∅

relevant(3) = (relevant(4) - DEF(3)) ∪ (REF(3) if relevant(4) ∩ DEF(3) ≠ ∅)
= ({b, d} - {d}) ∪ (∅ if {b, d} ∩ {d} ≠ ∅)
= {b}

Step 11: Calculate relevant(2)

Statement 2: c = 2
DEF(2) = {c}
REF(2) = ∅

relevant(2) = (relevant(3) - DEF(2)) ∪ (REF(2) if relevant(3) ∩ DEF(2) ≠ ∅)
= ({b} - {c}) ∪ (∅ if {b} ∩ {c} ≠ ∅)
= {b}

Step 12: Calculate relevant(1)

Statement 1: b = 1
DEF(1) = {b}
REF(1) = ∅

relevant(1) = (relevant(2) - DEF(1)) ∪ (REF(1) if relevant(2) ∩ DEF(1) ≠ ∅)
= ({b} - {b}) ∪ (∅ if {b} ∩ {b} ≠ ∅)
= ∅

Program #2

Solution:

n	Statement	REFs(n)	DEFs(n)	relevant(n)
1	#include <stdio.h>			
2	#include <math.h>			
3				
4	int main(void)			
5	{			
6	Double a, b, c, d, x1, x2;			
7	// Read input data			
8	printf("Enter the variables for the quadratic")			{a, b, c}
9	scanf("%lf%lf%lf", &a, &b, &c);	a, b, c	a, b, c	{a, b, c}
10				{a, b, c}
11	//Perform calculation			{a, b, c}
12	d=sqrt(b * b - 4. * a * c);	a, b, c	d	{a, b, c}
13	x1=(-b + d) / (2. * c);	b, d, c	x1	{b,d, a}
14	x2=(-b - d) / (2. * a);	b, d, a	x2	{b, d, a}
15				{x2}
16	//Display output			{x2}
17	printf("\nx1=%12.3e x2=%12.3e\n", x1,	x1, x2		{x2}

	x2);			
18	return 0;			
19	}			

Program Slice on <17, x2>:

{14, 12, 9}

n	**Statement**
9	scanf("%lf%lf%lf", &a, &b, &c);
12	d=sqrt(b * b - 4. * a * c);
14	x2=(-b - d) / (2. * a);

Program #2:

Detailed Calculation of relevant(n) for Slice <17, x2>

Initialization:

- **relevant(17) = {x2} (from the slicing criterion <17, x2>)**

Step 1: Calculate relevant(16)

Statement 16: //Display output
DEF(16) = ∅
REF(16) = ∅

relevant(16) = (relevant(17) - DEF(16)) ∪ (REF(16) if relevant(17) ∩ DEF(16) ≠ ∅)

= ({x2} - ∅) ∪ (∅ if {x2} ∩ ∅ ≠ ∅)
= {x2} ∪ ∅
= {x2}

Step 2: Calculate relevant(15)

Statement 15: (empty line)
DEF(15) = ∅
REF(15) = ∅

relevant(15) = (relevant(16) - DEF(15)) ∪ (REF(15) if relevant(16) ∩ DEF(15) ≠ ∅)
= ({x2} - ∅) ∪ (∅ if {x2} ∩ ∅ ≠ ∅)
= {x2}

Step 3: Calculate relevant(14)

Statement 14: x2 = (-b - d) / (2. * a)
DEF(14) = {x2}
REF(14) = {b, d, a}

relevant(14) = (relevant(15) - DEF(14)) ∪ (REF(14) if relevant(15) ∩ DEF(14) ≠ ∅)
= ({x2} - {x2}) ∪ ({b, d, a} if {x2} ∩ {x2} ≠ ∅)
= ∅ ∪ {b, d, a}
= {b, d, a}

Step 4: Calculate relevant(13)

Statement 13: x1 = (-b + d) / (2. * c)
DEF(13) = {x1}
REF(13) = {b, d, c}

relevant(13) = (relevant(14) - DEF(13)) ∪ (REF(13) if relevant(14) ∩ DEF(13) ≠ ∅)
= ({b, d, a} - {x1}) ∪ ({b, d, c} if {b, d, a} ∩ {x1} ≠ ∅)

= {b, d, a} ∪ ∅
= {b, d, a}

Step 5: Calculate relevant(12)

Statement 12: d = sqrt(b * b - 4. * a * c)
DEF(12) = {d}
REF(12) = {a, b, c}

relevant(12) = (relevant(14) - DEF(12)) ∪ (REF(12) if relevant(14) ∩ DEF(12) ≠ ∅)
= ({b, d, a} - {d}) ∪ ({a, b, c} if {b, d, a} ∩ {d} ≠ ∅)
= {b, a} ∪ {a, b, c}
= {a, b, c}

Step 6: Calculate relevant(11)

Statement 11: //Perform calculation
DEF(11) = ∅
REF(11) = ∅

relevant(11) = (relevant(12) - DEF(11)) ∪ (REF(11) if relevant(12) ∩ DEF(11) ≠ ∅)
= ({a, b, c} - ∅) ∪ (∅ if {a, b, c} ∩ ∅ ≠ ∅)
= {a, b, c}

Step 7: Calculate relevant(10)

Statement 10: (empty line)
DEF(10) = ∅
REF(10) = ∅

relevant(10) = (relevant(11) - DEF(10)) ∪ (REF(10) if relevant(11) ∩ DEF(10) ≠ ∅)
= ({a, b, c} - ∅) ∪ (∅ if {a, b, c} ∩ ∅ ≠ ∅)
= {a, b, c}

Step 8: Calculate relevant(9)

Statement 9: scanf("%lf%lf%lf", &a, &b, &c)
DEF(9) = {a, b, c}
REF(9) = {a, b, c}

relevant(9) = (relevant(10) - DEF(9)) ∪ (REF(9) if relevant(10) ∩ DEF(9) ≠ ∅)
= ({a, b, c} - {a, b, c}) ∪ ({a, b, c} if {a, b, c} ∩ {a, b, c} ≠ ∅)
= ∅ ∪ {a, b, c}
= {a, b, c}

Step 9: Calculate relevant(8)

Statement 8: printf("Enter the variables...")
DEF(8) = ∅
REF(8) = ∅

relevant(8) = (relevant(9) - DEF(8)) ∪ (REF(8) if relevant(9) ∩ DEF(8) ≠ ∅)
= ({a, b, c} - ∅) ∪ (∅ if {a, b, c} ∩ ∅ ≠ ∅)
= {a, b, c}

Steps 7-1: Earlier statements (1-7)

These are declarations and comments that don't affect the relevant sets.

Complete relevant sets for Program no. 2:

n	relevant(n)
17	{x2}
16	{x2}
15	{x2}
14	{b, d, a}
13	{b, d, a}

12	{a, b, c}
11	{a, b, c}
10	{a, b, c}
9	{a, b, c}
8	{a, b, c}
1-7	∅

Program no.3

Solution:

n	Statement	REFs(n)	DEFs(n)	relevant(n)
1	b=1		b	
2	c=2		c	b
3	d=5		d	b, c
4	a=3		a	b, c
5	While (a < 10)	a		b, c
6	b=b+c	b, c	b	b, c
7	c=c+1	c	c	b
8	a=b	b	a	b
9	EndWhile			a
10	print a	a		a

Program Slice on <10, a>:

{8, 7, 6, 2, 1}

n	Statement
1	b=1
2	c=2
6	b=b+c
7	c=c+1
8	a=b

Program no. 3: Detailed Calculation of relevant(n) for Slice <10, a>
Initialization:

- **relevant(10) = {a} (from the slicing criterion <10, a>)**

Step 1: Calculate relevant(9)

Statement 9: EndWhile
DEF(9) = ∅
REF(9) = ∅

relevant(9) = (relevant(10) - DEF(9)) ∪ (REF(9) if relevant(10) ∩ DEF(9) ≠ ∅)
= ({a} - ∅) ∪ (∅ if {a} ∩ ∅ ≠ ∅)
= {a}

Step 2: Calculate relevant(8)

Statement 8: a = b
DEF(8) = {a}
REF(8) = {b}

relevant(8) = (relevant(9) - DEF(8)) ∪ (REF(8) if relevant(9) ∩ DEF(8) ≠ ∅)
= ({a} - {a}) ∪ ({b} if {a} ∩ {a} ≠ ∅)
= ∅ ∪ {b}
= {b}

Step 3: Calculate relevant(7)

Statement 7: c = c + 1
DEF(7) = {c}
REF(7) = {c}

relevant(7) = (relevant(8) - DEF(7)) ∪ (REF(7) if relevant(8) ∩ DEF(7) ≠ ∅)
= ({b} - {c}) ∪ ({c} if {b} ∩ {c} ≠ ∅)
= {b} ∪ ∅
= {b}

Step 4: Calculate relevant(6)

Statement 6: b = b + c
DEF(6) = {b}
REF(6) = {b, c}

relevant(6) = (relevant(7) - DEF(6)) ∪ (REF(6) if relevant(7) ∩ DEF(6) ≠ ∅)
= ({b} - {b}) ∪ ({b, c} if {b} ∩ {b} ≠ ∅)
= ∅ ∪ {b, c}
= {b, c}

Step 5: Calculate relevant(5)

Statement 5: While (a < 10)
DEF(5) = ∅
REF(5) = {a}

relevant(5) = (relevant(6) - DEF(5)) ∪ (REF(5) if relevant(6) ∩ DEF(5) ≠ ∅)
= ({b, c} - ∅) ∪ ({a} if {b, c} ∩ ∅ ≠ ∅)
= {b, c}

Step 6: Calculate relevant(4)

Statement 4: a = 3
DEF(4) = {a}
REF(4) = ∅

relevant(4) = (relevant(5) - DEF(4)) ∪ (REF(4) if relevant(5) ∩ DEF(4) ≠ ∅)
= ({b, c} - {a}) ∪ (∅ if {b, c} ∩ {a} ≠ ∅)
= {b, c}

Step 7: Calculate relevant(3)

Statement 3: d = 5
DEF(3) = {d}
REF(3) = ∅

relevant(3) = (relevant(4) - DEF(3)) ∪ (REF(3) if relevant(4) ∩ DEF(3) ≠ ∅)
= ({b, c} - {d}) ∪ (∅ if {b, c} ∩ {d} ≠ ∅)
= {b, c}

Step 8: Calculate relevant(2)

Statement 2: c = 2
DEF(2) = {c}
REF(2) = ∅

relevant(2) = (relevant(3) - DEF(2)) ∪ (REF(2) if relevant(3) ∩ DEF(2) ≠ ∅)
= ({b, c} - {c}) ∪ (∅ if {b, c} ∩ {c} ≠ ∅)
= {b}

Step 9: Calculate relevant(1)

Statement 1: b = 1
DEF(1) = {b}
REF(1) = ∅

relevant(1) = (relevant(2) - DEF(1)) ∪ (REF(1) if relevant(2) ∩ DEF(1) ≠ ∅)
= ({b} - {b}) ∪ (∅ if {b} ∩ {b} ≠ ∅)
= ∅

Program no. 4

Solution:

n	Statement	REFs(n)	DEFs(n)	relevant(n)
1	read(text);	text		text
2	read(n);		n	text
3	lines=1;		lines	n, text
4	chars=1;		chars	n, text
5	subtext = ""		subtext	n, text
6	c= getChar(text);	text	c	Subtext ,n, text
7	while (c!= '\eof')	c		subtext ,c , n
8	If (c== '\n') then	c		subtext ,c , n

9	lines = lines + 1;	lines	lines	subtext ,c , n
10	chars = chars + 1;	chars	chars	subtext ,c , n
11	else chars=chars +1	chars	chars	subtext ,c , n
12	if(n!=0) then	n		subtext ,c , n
13	subtext = subtext ++ c	subtext, c	subtext	subtext ,c , n
14	n=n-1	n	n	subtext , n
15	c= getChar(text);	text	c	subtext
16	write(lines);	lines		subtext
17	write(chars);	chars		subtext
18	write(subtext);	subtext		subtext

Program Slice on <18, subtext>:

{13, 12, 14, 6, 2, 1, 5, 15}

n	**Statement**
1	read(text);
2	read(n);
5	subtext = ""
6	c= getChar(text);

12	if(n!=0) then
13	subtext = subtext ++ c
14	n=n-1
15	c= getChar(text);

Program 4:

Detailed Calculation of relevant(n) for Slice <18, subtext>

Initialization:

- **relevant(18) = {subtext} (from the slicing criterion <18, subtext>)**

Step 1: Calculate relevant(17)

Statement 17: write(chars)
DEF(17) = ∅
REF(17) = {chars}

relevant(17) = (relevant(18) - DEF(17)) ∪ (REF(17) if relevant(18) ∩ DEF(17) ≠ ∅)
= ({subtext} - ∅) ∪ (∅ if {subtext} ∩ ∅ ≠ ∅)
= {subtext}

Step 2: Calculate relevant(16)

Statement 16: write(lines)
DEF(16) = ∅
REF(16) = {lines}

relevant(16) = (relevant(17) - DEF(16)) ∪ (REF(16) if relevant(17) ∩ DEF(16) ≠ ∅)
= ({subtext} - ∅) ∪ (∅ if {subtext} ∩ ∅ ≠ ∅)
= {subtext}

Step 3: Calculate relevant(15)

Statement 15: c = getChar(text)
DEF(15) = {c}
REF(15) = {text}

relevant(15) = (relevant(16) - DEF(15)) ∪ (REF(15) if relevant(16) ∩ DEF(15) ≠ ∅)
= ({subtext} - {c}) ∪ ({text} if {subtext} ∩ {c} ≠ ∅)
= {subtext} ∪ ∅
= {subtext}

Step 4: Calculate relevant(14)

Statement 14: n = n - 1
DEF(14) = {n}
REF(14) = {n}

relevant(14) = (relevant(15) - DEF(14)) ∪ (REF(14) if relevant(15) ∩ DEF(14) ≠ ∅)
= ({subtext} - {n}) ∪ ({n} if {subtext} ∩ {n} ≠ ∅)
= {subtext} ∪ {n} (because subtext depends on n)
= {subtext, n}

Step 5: Calculate relevant(13)

Statement 13: subtext = subtext ++ c
DEF(13) = {subtext}
REF(13) = {subtext, c}

relevant(13) = (relevant(14) - DEF(13)) ∪ (REF(13) if relevant(14) ∩ DEF(13) ≠ ∅)
= ({subtext, n} - {subtext}) ∪ ({subtext, c} if {subtext, n} ∩ {subtext} ≠ ∅)
= {n} ∪ {subtext, c}
= {subtext, c, n}

Step 6: Calculate relevant(12)

Statement 12: if (n != 0) then
DEF(12) = ∅
REF(12) = {n}

relevant(12) = (relevant(13) - DEF(12)) ∪ (REF(12) if relevant(13) ∩ DEF(12) ≠ ∅)
= ({subtext, c, n} - ∅) ∪ ({n} if {subtext, c, n} ∩ ∅ ≠ ∅)
= {subtext, c, n}

Step 7: Calculate relevant(11)

Statement 11: else chars = chars + 1
DEF(11) = {chars}
REF(11) = {chars}

relevant(11) = (relevant(12) - DEF(11)) ∪ (REF(11) if relevant(12) ∩ DEF(11) ≠ ∅)
= ({subtext, c, n} - {chars}) ∪ (∅ if {subtext, c, n} ∩ {chars} ≠ ∅)
= {subtext, c, n}

Step 8: Calculate relevant(10)

Statement 10: chars = chars + 1
DEF(10) = {chars}
REF(10) = {chars}

relevant(10) = (relevant(11) - DEF(10)) ∪ (REF(10) if relevant(11) ∩ DEF(10) ≠ ∅)
= ({subtext, c, n} - {chars}) ∪ (∅ if {subtext, c, n} ∩ {chars} ≠ ∅)
= {subtext, c, n}

Step 9: Calculate relevant(9)

Statement 9: lines = lines + 1
DEF(9) = {lines}
REF(9) = {lines}

relevant(9) = (relevant(10) - DEF(9)) ∪ (REF(9) if relevant(10) ∩ DEF(9) ≠ ∅)
= ({subtext, c, n} - {lines}) ∪ (∅ if {subtext, c, n} ∩ {lines} ≠ ∅)
= {subtext, c, n}

Step 10: Calculate relevant(8)

Statement 8: If (c == '\n') then
DEF(8) = ∅
REF(8) = {c}

relevant(8) = (relevant(9) - DEF(8)) ∪ (REF(8) if relevant(9) ∩ DEF(8) ≠ ∅)
= ({subtext, c, n} - ∅) ∪ ({c} if {subtext, c, n} ∩ ∅ ≠ ∅)
= {subtext, c, n}

Step 11: Calculate relevant(7)

Statement 7: while (c != '\eof')
DEF(7) = ∅
REF(7) = {c}

relevant(7) = (relevant(8) - DEF(7)) ∪ (REF(7) if relevant(8) ∩ DEF(7) ≠ ∅)
= ({subtext, c, n} - ∅) ∪ ({c} if {subtext, c, n} ∩ ∅ ≠ ∅)
= {subtext, c, n}

Step 12: Calculate relevant(6)

Statement 6: c = getChar(text)
DEF(6) = {c}
REF(6) = {text}

relevant(6) = (relevant(7) - DEF(6)) ∪ (REF(6) if relevant(7) ∩ DEF(6) ≠ ∅)
= ({subtext, c, n} - {c}) ∪ ({text} if {subtext, c, n} ∩ {c} ≠ ∅)
= {subtext, n} ∪ {text}
= {subtext, n, text}

Step 13: Calculate relevant(5)

Statement 5: subtext = ""
DEF(5) = {subtext}
REF(5) = ∅

relevant(5) = (relevant(6) - DEF(5)) ∪ (REF(5) if relevant(6) ∩ DEF(5) ≠ ∅)
= ({subtext, n, text} - {subtext}) ∪ (∅ if {subtext, n, text} ∩ {subtext} ≠ ∅)
= {n, text}

Step 14: Calculate relevant(4)

Statement 4: chars = 1
DEF(4) = {chars}
REF(4) = ∅

relevant(4) = (relevant(5) - DEF(4)) ∪ (REF(4) if relevant(5) ∩ DEF(4) ≠ ∅)
= ({n, text} - {chars}) ∪ (∅ if {n, text} ∩ {chars} ≠ ∅)
= {n, text}

Step 15: Calculate relevant(3)

Statement 3: lines = 1
DEF(3) = {lines}
REF(3) = ∅

relevant(3) = (relevant(4) - DEF(3)) ∪ (REF(3) if relevant(4) ∩ DEF(3) ≠ ∅)
= ({n, text} - {lines}) ∪ (∅ if {n, text} ∩ {lines} ≠ ∅)
= {n, text}

Step 16: Calculate relevant(2)

Statement 2: read(n)
DEF(2) = {n}
REF(2) = ∅

relevant(2) = (relevant(3) - DEF(2)) ∪ (REF(2) if relevant(3) ∩ DEF(2) ≠ ∅)
= ({n, text} - {n}) ∪ (∅ if {n, text} ∩ {n} ≠ ∅)
= {text}

Step 17: Calculate relevant(1)

Statement 1: read(text)
DEF(1) = {text}
REF(1) = {text}

relevant(1) = (relevant(2) - DEF(1)) ∪ (REF(1) if relevant(2) ∩ DEF(1) ≠ ∅)
= ({text} - {text}) ∪ ({text} if {text} ∩ {text} ≠ ∅)
= ∅ ∪ {text}
= {text}

Program no. 5

Solution:

n	Statement	REFs(n)	DEFs(n)	relevant(n)
1	read(n);		n	
2	i=1;		i	
3	sum=0;		sum	i
4	product=1;		product	i
5	while (i < n) do	i, n		product , i
6	sum= sum +i;	sum, i	sum	Product , i
7	product= product *i	product, i	product	product , i
8	i=i +1;	i	i	Product

9	write(sum);	sum		product
10	write(product);	product		product

Program Slice on <10, product>:

{7, 5, 8, 4, 2, 1}

n	**Statement**
1	read(n);
2	i=1;
4	product=1;
5	while (i < n) do
7	product= product *i
8	i=i +1;

Program no. 5:
Detailed Calculation of relevant(n) for Slice <10,product>
Initialization:

- **relevant(10) = {product} (from the slicing criterion <10, product>)**

Step 1: Calculate relevant(9)

Statement 9: write(sum)
DEF(9) = ∅
REF(9) = {sum}

relevant(9) = (relevant(10) - DEF(9)) ∪ (REF(9) if relevant(10) ∩ DEF(9) ≠ ∅)
= ({product} - ∅) ∪ (∅ if {product} ∩ ∅ ≠ ∅)
= {product}

Step 2: Calculate relevant(8)

Statement 8: i = i + 1
DEF(8) = {i}
REF(8) = {i}

relevant(8) = (relevant(9) - DEF(8)) ∪ (REF(8) if relevant(9) ∩ DEF(8) ≠ ∅)
= ({product} - {i}) ∪ ({i} if {product} ∩ {i} ≠ ∅)
= {product} ∪ ∅
= {product}

Step 3: Calculate relevant(7)

Statement 7: product = product * i
DEF(7) = {product}
REF(7) = {product, i}

relevant(7) = (relevant(8) - DEF(7)) ∪ (REF(7) if relevant(8) ∩ DEF(7) ≠ ∅)
= ({product} - {product}) ∪ ({product, i} if {product} ∩ {product} ≠ ∅)
= ∅ ∪ {product, i}
= {product, i}

Step 4: Calculate relevant(6)

Statement 6: sum = sum + i
DEF(6) = {sum}
REF(6) = {sum, i}

relevant(6) = (relevant(7) - DEF(6)) ∪ (REF(6) if relevant(7) ∩ DEF(6) ≠ ∅)
= ({product, i} - {sum}) ∪ ({sum, i} if {product, i} ∩ {sum} ≠ ∅)
= {product, i} ∪ ∅
= {product, i}

Step 5: Calculate relevant(5)

Statement 5: while (i < n) do
DEF(5) = ∅
REF(5) = {i, n}

relevant(5) = (relevant(6) - DEF(5)) ∪ (REF(5) if relevant(6) ∩ DEF(5) ≠ ∅)
= ({product, i} - ∅) ∪ ({i, n} if {product, i} ∩ ∅ ≠ ∅)
= {product, i} ∪ ∅
= {product, i}

Step 6: Calculate relevant(4)

Statement 4: product = 1
DEF(4) = {product}
REF(4) = ∅

relevant(4) = (relevant(5) - DEF(4)) ∪ (REF(4) if relevant(5) ∩ DEF(4) ≠ ∅)
= ({product, i} - {product}) ∪ (∅ if {product, i} ∩ {product} ≠ ∅)
= {i} ∪ ∅
= {i}

Step 7: Calculate relevant(3)

Statement 3: sum = 0
DEF(3) = {sum}
REF(3) = ∅

relevant(3) = (relevant(4) - DEF(3)) ∪ (REF(3) if relevant(4) ∩ DEF(3) ≠ ∅)
= ({i} - {sum}) ∪ (∅ if {i} ∩ {sum} ≠ ∅)
= {i}

Step 8: Calculate relevant(2)

Statement 2: i = 1
DEF(2) = {i}
REF(2) = ∅

relevant(2) = (relevant(3) - DEF(2)) ∪ (REF(2) if relevant(3) ∩ DEF(2) ≠ ∅)
= ({i} - {i}) ∪ (∅ if {i} ∩ {i} ≠ ∅)
= ∅ ∪ ∅
= ∅

Step 9: Calculate relevant(1)

Statement 1: read(n)
DEF(1) = {n}
REF(1) = ∅

relevant(1) = (relevant(2) - DEF(1)) ∪ (REF(1) if relevant(2) ∩ DEF(1) ≠ ∅)
= (∅ - {n}) ∪ (∅ if ∅ ∩ {n} ≠ ∅)
= ∅