

# Subject Name (Gujarati)

4331105 -- Winter 2024

Semester 1 Study Material

Detailed Solutions and Explanations

## પ્રશ્ન 1(a) [3 ગુણ]

C લેંગેજના કોઈ પણ છ કીવર્ડ લખો.

### જવાબ

વર્ગ	કીવર્ડ
ડેટા ટાઇપ્સ	int, float, char
કંટ્રોલ ફલો	if, for, return

### મેમરી ટ્રીક

"I Find Clever Reasons For Results"

## પ્રશ્ન 1(b) [4 ગુણ]

ઓપરેટરની વ્યાખ્યા આપી operand ના આધારે તેના પ્રકાર જણાવો.

### જવાબ

ઓપરેટર: એવું ચિહ્ન કે જે ઓપરન્ડ્સ પર કામ કરીને પરિણામ આપે છે.

પ્રકાર	વિગત	ઉદાહરણ
ચુનરી	એક ઓપરન્ડ	++, --, !
બાઇનરી	બે ઓપરન્ડ	, -, *, /, %
ટર્નરી	ત્રણ ઓપરન્ડ	?:

### મેમરી ટ્રીક

"U-B-T: Use Binary Then Ternary"

## પ્રશ્ન 1(c) [7 ગુણ]

ફલોચાર્ટની વ્યાખ્યા લખો. ફલોચાર્ટના સિમ્બોલ દોરો. બે પૂણીક નંબર N1 અને N2 માંથી નાનો નંબર શોધવા માટેનો ફલોચાર્ટ દોરો.

### જવાબ

ફલોચાર્ટ: અલ્ગોરિધમનું ગ્રાફિકલ નિરૂપણ જેમાં પ્રમાણિત ચિહ્નો દ્વારા ક્રમબદ્ધ ઓપરેશનો દર્શાવવામાં આવે છે.  
ફલોચાર્ટના સામાન્ય ચિહ્નો:

ચિહ્ન	અર્થ
ઓવલ	શરૂઆત/અંત
પેરાલેલોગ્રામ	ઇનપુટ/આઉટપુટ
લંબચોરસ	પ્રક્રિયા
ડાયમંડ	નિર્ણય
એરો	ફલો દિશા

N1 અને N2 માંથી નાનો નંબર શોધવા માટેનો ફ્લોયાર્ડ:

```
flowchart LR
    A([Start]) --> B[/Input N1, N2/]
    B --> C{N1 < N2?}
    C -- Yes --> D[Min = N1]
    C -- No --> E[Min = N2]
    D --> F[/Display Min/]
    E --> F
    F --> G([End])
```

મેમરી ટ્રીક

“SPADE: Start-Process-Arrow-Decision-End”

### પ્રશ્ન 1(c) OR [7 ગુણ]

અલગોરિધમની વ્યાખ્યા લખો. વર્તુળનું ક્ષેત્રફળ અને પરિધિ શોધવા માટેનો અલગોરિધમ લખો.

જવાબ

અલગોરિધમ: કોઈ સમસ્યાને ઉકેલવા માટેનું પગલે પગલે પદ્ધતિસરનું કાર્યવાહી, જેમાં સુનિશ્ચિત સંખ્યામાં સુવ્યાખ્યાચિત સૂચનાઓનો ઉપયોગ થાય છે.

વર્તુળનું ક્ષેત્રફળ અને પરિધિ શોધવા માટેનો અલગોરિધમ:

1. શરૂઆત
2. નિઝયા  $r$  ઇનપુટ કરો
3.  $PI = 3.14159$  રેટ કરો
4.  $\text{ક્ષેત્રફળ} = PI \times r \times r$
5.  $\text{પરિધિ} = 2 \times PI \times r$
6. ક્ષેત્રફળ અને પરિધિ દર્શાવો
7. અંત

ઉપયોગિતા સૂત્રોનો કોષ્ટક:

માપન	સૂત્ર
ક્ષેત્રફળ	$\pi \times r^2$
પરિધિ	$2 \times \pi \times r$

મેમરી ટ્રીક

“RICARD: Radius Input, Calculate Area, Reveal Dimensions”

### પ્રશ્ન 2(a) [3 ગુણ]

printf() અને scanf() વર્ણણો તફાવત સમજાવો.

જવાબ

લક્ષણ	printf()	scanf()
હેતુ	સ્ક્રીન પર ડેટા આઉટપુટ કરે	ક્રીબોર્ડથી ડેટા ઇનપુટ કરે
દિશા	આઉટપુટ ફુકશન	ઇનપુટ ફુકશન
ફોર્મેટ સ્પેસિફિકર	જરૂરી	જરૂરી
પેરામીટર	વાસ્તવિક મૂલ્યો	ચલનું સરનામું (&)

મેમરી ટ્રીક

“OIAD: Output-Input, Actual-Destination”

## પ્રશ્ન 2(b) [4 ગુણ]

1 થી n સુધીનો સરવાળો અને સરેરાશ પ્રિન્ટ કરવા માટે પ્રોગ્રામ C લખો.

### જવાબ

```
\#include <stdio.h>

int main() {
    int n, i, sum = 0;
    float avg;

    printf("Enter n: ");
    scanf("\%d", &n);

    for(i = 1; i {=} n; i++) {
        sum += i;
    }

    avg = (float)sum / n;

    printf("Sum = \%d{n}", sum);
    printf("Average = \%.2f{n}", avg);

    return 0;
}
```

### મુખ્ય બિંદુઓ:

- ઇનિશિયલાઇઝેશન: sum = 0
- પુનરાવર્તન: 1 થી n સુધી for લૂપ
- ટાઇપ કાસ્ટિંગ: સાચા સરેરાશ માટે (float)

### મેમરી ટ્રીક

“SIAP: Sum Initialize, Add in loop, Print results”

## પ્રશ્ન 2(c) [7 ગુણ]

એરિથમેટિક ઓપરેટર અને રિલેશનલ ઓપરેટરને ઉદાહરણ સાથે સમજાવો.

### જવાબ

#### 1. એરિથમેટિક ઓપરેટર:

ઓપરેટર	ઓપરેશન	ઉદાહરણ	પરિણામ
+	સરવાળો	5 + 3	8
-	બાદબાકી	5 - 3	2
*	ગુણાકાર	5 * 3	15
/	ભાગાકાર	5 / 2	2 (પૂણાર્થ)
%	મોડ્યુલો (શેષ)	5 % 2	1

## 2. રિલેશનલ ઓપરેટર:

ઓપરેટર	અર્થ	ઉદાહરણ	પરિણામ
<	કરતાં નાનું	5 < 3	0 (ખોટું)
>	કરતાં મોટું	5 > 3	1 (સાચું)
<=	નાનું અથવા સમાન	5 <= 5	1 (સાચું)
>=	મોટું અથવા સમાન	3 >= 5	0 (ખોટું)
==	સમાન	5 == 5	1 (સાચું)
!=	અસમાન	5 != 3	1 (સાચું)

### કોડ ઉદાહરણ:

```
int
a = 5,
b = 3;
printf("a + b = %d\n", a + b);      //    : 8
printf("a { b is }%d\n", a {} b);   //    : 1 ( )
```

### મેમરી ટ્રીક

“ASMDR for Arithmetic, LEGENE for Relational”

## પ્રશ્ન 2(a) OR [3 ગુણ]

get(S) અને scanf(`%s",S) વચ્ચેનો તફાવત સમજાવો.

### જવાબ

લક્ષણ	gets(S)	scanf(`%s",S)
વહાઇટસ્પેસ હેન્ડલિંગ	સ્પેસ વાંચે છે	વહાઇટસ્પેસ પર બંધ થાય છે
બફર ઓવરફલો	બાઉન્ડરી ચેક નથી	વિડુથ લિમિટ સાથે સલામત
રિટર્ન ટાઇપ	char*	વાંચેલા આઇટમની સંખ્યા
ઉપયોગ સુરક્ષા	ડેપ્રિકેટ, અસુરક્ષિત	ફોર્મેટ કંટ્રોલ સાથે સલામત

### મેમરી ટ્રીક

“WBRU: Whitespace-Boundary-Return-Usage”

## પ્રશ્ન 2(b) OR [4 ગુણ]

એ નંબરોના મૂલ્યની અદલાબદલી (એક્સચેન્જ) કરવા માટે C પ્રોગ્રામ લખો.

### જવાબ

```
\#include <stdio.h>

int main() {
    int a, b, temp;

    printf("Enter two numbers: ");
    scanf("%d %d", &a, &b);

    printf("Before swap:
```

```

a = \%d,
b = \%d{n}", a, b);

temp = a;
a = b;
b = temp;

printf("After swap:

a = \%d,
b = \%d{n}", a, b);

return 0;
}

```

### ડાયાગ્રામ:

```

flowchart LR
    subgraph Before
        A1[a = 5] --> B1[b = 10]
        end
    subgraph Step1
        A2[a = 5] --> T1[temp = 5]
        B2[b = 10]
        end
    subgraph Step2
        A3[a = 10] --> T2[temp = 5]
        B3[b = 10]
        end
    subgraph After
        A4[a = 10] --> B4[b = 5]
        end
    Before --> Step1
    Step1 --> Step2
    Step2 --> After

```

### મેમરી ટ્રીક

``TAB: Temp-Assign-Backfill''

### પ્રશ્ન 2(c) OR [7 ગુણ]

બુલિયન ઓપરેટર અને લોજિકલ ઓપરેટર ઉદાહરણ સાથે સમજાવો.

### જવાબ

#### 1. બુલિયન ઓપરેટર:

ઓપરેટર	ઓપરેશન	ઉદાહરણ	પરિણામ
&	બિટવાઈઝ AND	5 & 3	1
	બિટવાઈઝ OR	5   3	7
^	બિટવાઈઝ XOR	5 ^ 3	6
~	બિટવાઈઝ NOT	~5	-6
<<	લેફ્ટ શિફ્ટ	5 << 1	10
>>	રાઇટ શિફ્ટ	5 >> 1	2

## 2. લોજિકલ ઓપરેટર:

ઓપરેટર	અર્થ	ઉદાહરણ	પરિણામ
&&	લોજિકલ AND	(5>3) && (2<4)	1 (સાચું)
	લોજિકલ OR	(5<3)    (2<4)	1 (સાચું)
!	લોજિકલ NOT	!(5>3)	0 (ખોટું)

### ઉદાહરણ:

```
int
a = 5,
b = 3;

printf("a & b = %d\n", a & b);           //      : 1 (      AND)
printf("a { b && b 10 is }%d\n", a {} b && b {} 10); //      : 1 ( )
```

### બિટ રિપોન્ટેશન (5 & 3):

```
5 = 101
3 = 011
& = 001 (1 in decimal)
```

### મેમરી ટ્રીક

“BOXNRL for Boolean, AON for Logical”

## પ્રશ્ન 3(a) [3 ગુણ]

Entry controlled અને exit controlled લૂપની ઉદાહરણ સાથે સરખામણી કરો.

### જવાબ

લક્ષણ	Entry Controlled	Exit Controlled
શરત ચકાસણી	અમલ પહેલા	અમલ પછી
ન્યૂનતમ પુનરાવર્તન	શૂન્ય	એક
ઉદાહરણ	while, for	do-while
ઉપયોગ	પ્રી-ચેક જરૂરી હોય	ઓછામાં ઓછું એક વાર અમલ જરૂરી હોય

### મેમરી ટ્રીક

“BCME: Before-Check-Multiple-Examples”

## પ્રશ્ન 3(b) [4 ગુણ]

સ્વીચ કેસનો ઉપયોગ કરીને બે નંબરોના સરવાળા અને બાદબાકી દર્શાવવા માટેનો C પ્રોગ્રામ લખો.

### જવાબ

```
#include <stdio.h>

int main() {
    int a, b, choice, result;

    printf("Enter two numbers: ");
    scanf("%d %d", &a, &b);
```

```

printf("1. Addition\n2. Subtraction");
printf("Enter choice (1/2): ");
scanf("\%d", &choice);

switch(choice) {
    case 1:
        result = a + b;
        printf("Addition: %d\n", result);
        break;
    case 2:
        result = a - b;
        printf("Subtraction: %d\n", result);
        break;
    default:
        printf("Invalid choice\n");
}
}

return 0;
}

```

### ફ્લોચાર્ટ:

```

flowchart LR
    A([Start]) --> B[/Input a, b/]
    B --> C[/Input choice/]
    C --> D{"switch(choice)"}
    D --> E[result = a + b]
    D --> F[result = a - b]
    D --> G[Display Invalid]
    E --> H[/Display Addition/]
    F --> I[/Display Subtraction/]
    H --> J([End])
    I --> J
    G --> J

```

### મેમરી ટ્રીક

“CIRCA: Choice-Input-Result-Calculate-Action”

### પ્રશ્ન 3(c) [7 ગુણ]

સિન્ટેક્સ, ફ્લોચાર્ટ અને ઉદાહરણ સાથે multiple if-else સ્ટેટમેન્ટ સમજાવો.

#### જવાબ

##### multiple if-else સિન્ટેક્સ:

```

if (condition1) {
    // code block 1
}
else if (condition2) {
    // code block 2
}
else if (condition3) {
    // code block 3
}
else {
    // default code block
}

```

### ફ્લોચાર્ટ:

```

flowchart LR
    A([Start]) --{-}--> B\{condition1?\}
    B --{-}--> C[Execute block 1]
    B --{-}--> D\{condition2?\}
    D --{-}--> E[Execute block 2]
    D --{-}--> F\{condition3?\}
    F --{-}--> G[Execute block 3]
    F --{-}--> H[Execute default block]
    C --{-}--> I([End])
    E --{-}--> I
    G --{-}--> I
    H --{-}--> I

```

### ઉદાહરણ:

```

#include <stdio.h>

int main() {
    int marks;

    printf("Enter marks: ");
    scanf("%d", &marks);

    if (marks == 80) {
        printf("Grade: A{n}");
    }
    else if (marks == 70) {
        printf("Grade: B{n}");
    }
    else if (marks == 60) {
        printf("Grade: C{n}");
    }
    else {
        printf("Grade: F{n}");
    }

    return 0;
}

```

### મેમરી ટ્રીક

“TEST: Try Each Statement Then default”

### પ્રશ્ન 3(a) OR [3 ગુણ]

**break** અને **continue** કીવર્કનો ઉપયોગ જણાવો.

#### જવાબ

કીવર્ક	હેતુ	અસર	સામાન્ય ઉપયોગ
<b>break</b>	લૂપ/સ્વિચ ટમ્ભિનેટ કરે	વર્તમાન લૂપ/સ્વિચ છોડી દે	શરત પૂરી થાય ત્યારે બહાર નીકળવા
<b>continue</b>	ઇટરેશન છોડો	આગામી ઇટરેશન પર જાય	ચોક્કસ મૂલ્યો છોડવા

### ઉદાહરણ કોડ:

```
// break
for(i=1; i{=}10; i++) \{
if(i == 5) break; //  
  
i=5  
  
    printf("\%d ", i); // 1 2 3 4  
\}  
  
// continue
for(i=1; i{=}5; i++) \{
if(i == 3) continue; //  
  
i=3  
  
    printf("\%d ", i); // 1 2 4 5  
\}
```

### મેમરી ટ્રીક

``EXIT-SKIP: EXit IT or SKIP iteration''

### પ્રશ્ન 3(b) OR [4 ગુણ]

આપેલ સંખ્યા એકી છે કે બેકી તે ચકાસવા માટે C પ્રોગ્રામનો લખો.

#### જવાબ

```
\#include <stdio.h>  
  
int main() \{
    int num;  
  
    printf("Enter a number: ");
    scanf("\%d", &num);  
  
    if (num \% 2 == 0) \{
        printf("\%d is even.\n", num);
    \}
    else \{
        printf("\%d is odd.\n", num);
    \}  
  
    return 0;
\}
```

#### ડાયાગ્રામ:

```
flowchart LR
    A([Start]) --> B[/Input num/]
    B --> C{num \% 2 == 0?}
    C -- Yes --> D[/Display "Even"]
    C -- No --> E[/Display "Odd"]
    D --> F([End])
    E --> F
```

#### મુખ્ય ટિપ્પણી:

- ચકાસણી: મોડ્યુલો (%) ઓપરેટરનો ઉપયોગ
- નિણય: 2 સાથેના શેષ પર આધારિત
- આઉટપુટ: શેષ 0 હોય તો બેકી, અન્યથા એકી

“MODE: MODulo Equals zero for even”

### પ્રશ્ન 3(c) OR [7 ગુણ]

સિન્ટેક્સ, ફ્લોચાર્ટ અને ઉદાહરણ સાથે switch-case statement સ્ટેપેન્ટ સમજાવો.

#### જવાબ

##### switch-case સિન્ટેક્સ:

```
switch (expression) \{
    case constant1:
        // code block 1
        break;
    case constant2:
        // code block 2
        break;
    ...
    default:
        // default code block
\}
```

##### ફ્લોચાર્ટ:

```
flowchart LR
    A([Start]) --> B[Evaluate expression]
    B --> C{matches case 1?}
    C -- Yes --> D[Execute block 1]
    C -- No --> E{matches case 2?}
    E -- Yes --> F[Execute block 2]
    E -- No --> G{matches case n?}
    G -- Yes --> H[Execute block n]
    G -- No --> I[Execute default block]
    D --> J{break?}
    F --> K{break?}
    H --> L{break?}
    J -- Yes --> M([End])
    K -- Yes --> M
    L -- Yes --> M
    I --> M
    J -- No --> E
    K -- No --> G
    L -- No --> M
```

##### ઉદાહરણ:

```
\#include <stdio.h>

int main() \{
    char grade;

    printf("Enter grade (A{-D): ");
    scanf(" %c", \&grade);

    switch (grade) \{
        case {A}:
            printf("Excellent!\n");
            break;
        case {B}:
            printf("Good job!\n");
            break;
    \}
```

```

        case {C}:
            printf("Satisfactory{n}");
            break;
        case {D}:
            printf("Needs improvement{n}");
            break;
        default:
            printf("Invalid grade{n}");
    }

    return 0;
}

```

### મેમરી ટ્રીક

“CEBID: Compare-Execute-Break-If-Done”

### પ્રશ્ન 4(a) [3 ગુણ]

સ્ટ્રિંગ વ્યાખ્યાપિત કરો. સ્ટ્રિંગ પર કરી શકાય તેવા વિવિધ ઓપરેશન્સની યાદી આપો.

#### જવાબ

સ્ટ્રિંગ: કેરેક્ટર્સનો એરે જે NULL કેરેક્ટર '10' થી સમાપ્ત થાય છે.

ઓપરેશન	વિગત	ફંક્શન
ઇનપુટ/આઉટપુટ	સ્ટ્રિંગ વાંચવી/લખવી	gets(), puts()
કોપી	એક સ્ટ્રિંગને બીજું સ્ટ્રિંગમાં કોપી કરવી	strcpy()
જોડાણ	બે સ્ટ્રિંગ જોડવી	strcat()
સરખામણી	બે સ્ટ્રિંગની સરખામણી	strcmp()
લંબાઈ	સ્ટ્રિંગની લંબાઈ શોધવી	strlen()
શોધ	સબસ્ટ્રિંગ શોધવી	strstr()

### મેમરી ટ્રીક

“ICCLS: Input-Copy-Concatenate-Length-Search”

### પ્રશ્ન 4(b) [4 ગુણ]

અપરકેસ આલ્ફાબેટને લોઅરકેસ આલ્ફાબેટમાં કન્વર્ટ કરવા માટે C પ્રોગ્રામ લખો.

#### જવાબ

```

#include <stdio.h>

int main() {
    char ch;

    printf("Enter an uppercase letter: ");
    scanf(" %c", &ch);

    if (ch == 'A' || ch == 'Z') {
        char lowercase = ch + 32; // ASCII 32
        printf("Lowercase: %c\n", lowercase);
    } else {
        printf("Not an uppercase letter\n");
    }
}

```

```
    return 0;  
}
```

ASCII ટેબલનો અંશ:

કોડ	ASCII મુદ્દ્ય
A	65
a	97
Z	90
z	122
તફાવત	32

## મેમરી ટ્રીક

“COOL: Character Offset Of Lowercase”

### પ્રશ્ન 4(c) [7 ગુણ]

for લૂપ માટેનો ફ્લોચાર્ટ દોરો અને ઉદાહરણ સાથે સમજાવો.

#### જવાબ

##### For Loop સિન્ક્રેક્ષન:

```
for (initialization; condition; increment/decrement) {  
    // code block  
}
```

##### ફ્લોચાર્ટ:

```
graph TD  
    A([Start]) --> B[Initialization]  
    B --> C{Condition?}  
    C -- True --> D[Execute code block]  
    D --> E[Increment/Decrement]  
    E --> F{[End]}  
    F -- False --> C
```

##### For Loop ઘટકો:

- ઇનિશિયલાઇઝેશન: શરૂઆતમાં એક વખત અમલ
- શરત: દરેક પુનરાવર્તન પહેલાં ચકાસાય
- વૃદ્ધિ/ઘટાડો: દરેક પુનરાવર્તન પછી અમલ
- બોડી: શરત સાચી હોય તો અમલ

##### ઉદાહરણ:

```
\#include <stdio.h>  
  
int main() {  
    int i;  
  
    for (i = 1; i <= 5; i++) {  
        printf("\%d ", i);  
    }  
    // : 1 2 3 4 5  
  
    return 0;  
}
```

##### અમલીકરણનો પ્રવાહ:

- i = 1 પ્રારંભિક
- શરત ચકાસો (1 <= 5) - સાચું

3. બોડી અમલ - 1 પ્રિન્ટ
4. i ને 2 માં વૃદ્ધિ
5. શરત ચકાસો (2 <= 5) - સાચું
6. અને એ રીતે આગળ i = 6 થાય ત્યાં સુધી

### મેમરી ટ્રીક

“ICE-T: Initialize, Check, Execute, Then increment”

### પ્રશ્ન 4(a) OR [3 ગુણ]

એરે વ્યાખ્યાયિત કરો. એરે પર કરી શકાય તેવા વિવિધ ઓપરેશન્સની યાદી આપો.

#### જવાબ

એરે: સમાન ડેટા પ્રકારના તત્ત્વો જે કમિક મેમરી સ્થાનોમાં સંગ્રહિત થાય છે.

ઓપરેશન	વિગત	ઉદાહરણ
ઘોષણા	એરે બનાવો	int arr[5];
ઇનિશિયલાઇઝેશન	મૂલ્ય આપો	arr[0] = 10;
ટ્રેવર્સલ	તમામ તત્ત્વો એક્સેસ કરો	for loop
ઇન્સર્શન	નવું તત્ત્વ ઉમેરો	arr[pos] = value;
ડિલીશન	તત્ત્વ ફૂર કરો	તત્ત્વો શિક્ષિત કરો
સર્ચિંગ	તત્ત્વ શોધો	લિનિયર/બાઇનરી શોધ
સોર્ટિંગ	તત્ત્વો ગોઠવો	બબલ/સિલેક્શન સોર્ટ

### મેમરી ટ્રીક

“DITIDSS: Declare-Initialize-Traverse-Insert-Delete-Search-Sort”

### પ્રશ્ન 4(b) OR [4 ગુણ]

pointer વ્યાખ્યાયિત કરો. ઉદાહરણ સાથે સમજાવો.

#### જવાબ

પોઇન્ટર: એવું ચલ જે અન્ય ચલના મેમરી સરનામાને સંગ્રહિત કરે છે.

કન્સેપ્ટ	વિગત	સિન્ટેક્સ
ઘોષણા	પોઇન્ટર બનાવો	int *ptr;
એક્સેસ ઓપરેટર	સરનામું મેળવો	&variable
ડિરેક્ટરન્સિંગ	સરનામા પર મૂલ્ય એક્સેસ કરો	*ptr
ઓસાઇનમેન્ટ	પોઇન્ટરમાં સરનામું સંગ્રહી	ptr = &variable;

### ઉદાહરણ:

```
\#include <stdio.h>

int main() {
    int num = 10;
    int *ptr;

    ptr = &num; // num      ptr

    printf("Value of num: %d\n", num);           // 10
    printf("Address of num: %p\n", &num);         // num
    printf("Value of ptr: %p\n", ptr);             //
    printf("Value pointed by ptr: %d\n", *ptr);   // 10

    *ptr = 20; // 
    printf("New value of num: %d\n", num);         // 20

    return 0;
}
```

### ડાયગ્રામ:

```
flowchart LR
    subgraph Memory
        direction LR
        A["A[ptr: 1001] {---|contains| B[1000]}"]
        C["C[num: 1000] {---|contains| D[10]}"]
        end
        B["B {-.{-}|points to| C}"]
```

### મેમરી ટ્રીક

``SAVD: Store Address, Value through Dereferencing''

## પ્રશ્ન 4(c) OR [7 ગુણ]

while લૂપ માટેનો ફ્લોચાર્ટ દોરો અને ઉદાહરણ સાથે સમજાવો.

### જવાબ

#### While Loop સિન્ક્રિપ્શન:

```
while (condition) {
    // code block
}
```

#### ફ્લોચાર્ટ:

```
flowchart LR
    A([Start]) {---|True| B\{Condition?\}}
    B {---|True| C[Execute code block]}
    C {---|---| D[Update control variable]}
    D {---|---| B}
    B {---|False| E([End])}
```

#### While Loop ઘટકો:

- ઇનિશિયલાઇઝેશન: લૂપની પહેલાં
- શરત: દરેક પુનરાવર્તન પહેલાં ચકાસાય
- બોડી: શરત સાચી હોય તો અમલ
- અપડેટ: બોડીની અંદર કરવું જરૂરી

### ઉદાહરણ:

```
\#include <stdio.h>

int main() {
    int i = 1;

    while (i <= 5) {
        printf("\%d ", i);
        i++;
    }
    // : 1 2 3 4 5

    return 0;
}
```

#### અમલીકરણનો પ્રવાહ:

1. i = 1 પ્રારંભિક (લૂપ પહેલાં)
2. શરત ચકાસો (1 <= 5) - સાચું
3. બોડી અમલ - 1 પ્રિન્ટ
4. i ને 2 માં અપડેટ
5. શરત ચકાસો (2 <= 5) - સાચું
6. અને એ રીતે આગામિ i = 6 થાય ત્યાં સુધી

#### મેમરી ટ્રીક

“CHECK-UPDATE: CHECK before entering, UPDATE before repeating”

#### પ્રશ્ન 5(a) [3 ગુણ]

નીચેના functions નો ઉપયોગ જણાવો. (1) strcat() (2) strlen() (3) strcpy()

#### જવાબ

ફંક્શન	હેતુ	સિન્ટેક્સ	ઉદાહરણ
strcat()	સિંગારસ જોડે છે	strcat(dest, src)	“Hello” + “World” -> “HelloWorld”
strlen()	સિંગાની લંબાઈ આપે છે	strlen(str)	“Hello” -> 5
strcpy()	સિંગા કોપી કરે છે	strcpy(dest, src)	src -> dest

#### કોડ ઉદાહરણ:

```
\#include <string.h>

char str1[20] = "Hello";
char str2[20] = "World";
char str3[20];

strcat(str1, str2); // str1 "HelloWorld"
int len = strlen(str1); // len 10
strcpy(str3, str1); // str3 "HelloWorld"
```

#### મેમરી ટ્રીક

“CLS: Concatenate-Length-Source copy”

#### પ્રશ્ન 5(b) [4 ગુણ]

પુસ્તકની માહિતી સંગ્રહિત કરવા માટે એક સ્ટ્રક્ચર બનાવો: book\_no, book\_title, book\_author, book\_price.

## જવાબ

```

#include <stdio.h>
#include <string.h>

struct Book {
    int book_no;
    char book_title[50];
    char book_author[30];
    float book_price;
};

int main() {
    struct Book book1;

    //
    book1.book_no = 101;
    strcpy(book1.book_title, "Programming in C");
    strcpy(book1.book_author, "Dennis Ritchie");
    book1.book_price = 450.75;

    //
    printf("Book No: %d\n", book1.book_no);
    printf("Title: %s\n", book1.book_title);
    printf("Author: %s\n", book1.book_author);
    printf("Price: %.2f\n", book1.book_price);

    return 0;
}

```

## સ્ક્રીપ્ટ મેમરી લેઆઉટ:

```

graph TD
    subgraph "struct Book"
        direction LR
        A[book_no: 4 bytes] --- B[book_title: 50 bytes]
        B --- C[book_author: 30 bytes]
        C --- D[book_price: 4 bytes]
    end

```

## મેમરી ટ્રીક

“NTAP: Number-Title-Author-Price”

## પ્રશ્ન 5(c) [7 ગુણ]

એરે અને એરે initialization સમજાવો. ઉદાહરણ આપો.

## જવાબ

એરે: સમાન ડેટા પ્રકારના તત્ત્વોનો સમૂહ જે કમિક મેમરી સ્થાનોમાં સંગ્રહિત થાય છે.

એરે ઇનિશિયલાઇઝેશન પદ્ધતિઓ:

પદ્ધતિ	સિન્ટેક્સ	ઉદાહરણ
ઘોષણા સમયે	datatype array_name[size] = {values};	int arr[5] = {10, 20, 30, 40, 50};
આંશિક	datatype array_name[size] = {values};	int arr[5] = {10, 20}; // બાકીના 0 છે
બધા શૂન્ય	datatype array_name[size] = {0};	int arr[5] = {0};
તત્ત્વ દર તત્ત્વ	array_name[index] = value;	arr[0] = 10; arr[1] = 20;

સાઈઝ અનુમાન

```
datatype array_name[] = int arr[] = {10, 20, 30}; // સાઈઝ 3  
{values};
```

ઉદાહરણ:

```
\#include <stdio.h>

int main() {
    // 1:
    int arr1[5] = {10, 20, 30, 40, 50};

    // 2: ( 0 )
    int arr2[5] = {10, 20}; // arr2[2], arr2[3], arr2[4]

    // 3:
    int arr3[3];
    arr3[0] = 100;
    arr3[1] = 200;
    arr3[2] = 300;

    // 4:
    int arr4[] = {1, 2, 3, 4, 5}; // 5

    //
    printf("arr1[2] = %d\n", arr1[2]); // : 30

    //
    printf("arr4 elements: ");
    for(int i = 0; i < 5; i++) {
        printf("%d ", arr4[i]); // : 1 2 3 4 5
    }

    return 0;
}
```

મેમરી રિપોર્ટેશન:

```
flowchart LR
    subgraph "arr1[5]"
    direction LR
    A["arr1[0]: 10"] --> B["arr1[1]: 20"]
    B --> C["arr1[2]: 30"]
    C --> D["arr1[3]: 40"]
    D --> E["arr1[4]: 50"]
    end
```

મેમરી ટ્રીક

``CAPES: Complete, Automatic, Partial, Element, Size-inferred''

### પ્રશ્ન 5(a) OR [3 ગુણ]

array અને structure ઉદાહરણ સાથે સરખામણી કરો.

જવાબ

લક્ષણ	Array	Structure
ડેટા ટાઇપ	સમાન પ્રકારના તત્ત્વો	અલગ પ્રકારના તત્ત્વો
એક્સેસ	ઇન્ડેક્સનો ઉપયોગ (arr[i])	ડોટ ઓપરેટરનો ઉપયોગ (s.member)
મેમરી	કમિક, ફિક્સ સાઈઝ	કમિક, પેરિંગ હોઇચ શકે
અસાઇનમેન્ટ	તત્વ દર તત્વ	સીધું જ સુસંગત રટ્કચર સાથે
હેતુ	સમાન આઇટમોનો સંગ્રહ	સંબંધિત ડેટાનો સમૂહ

### અરે ઉદાહરણ:

```
int marks[5] = \{85, 90, 78, 92, 88\;}
printf("\%d", marks[2]); //      2 (78)
```

### સ્ટ્રક્ચર ઉદાહરણ:

```
struct Student \{
    int roll_no;
    char name[20];
    float percentage;
\;}

struct Student s1 = \{101, "Raj", 85.5\;};
printf("\%s", s1.name); // name ("Raj")
```

### મેમ્સી ટ્રીક

“DAMPA: Datatype-Access-Memory-Purpose-Assignment”

## પ્રશ્ન 5(b) OR [4 ગુણ]

User Defined Function વ્યાખ્યાપિત કરો. ઉદાહરણ સાથે સમજાવો.

### જવાબ

**User Defined Function:** પ્રોગ્રામર દ્વારા લખાયેલ કોડનો બ્લોક જે ચોક્કસ કાર્ય કરે છે અને તેને વારંવાર કોલ કરી શકાય છે.

ઘટક	વિગત	ઉદાહરણ
રિટર્ન ટાઇપ	પરત થતા ડેટાનો પ્રકાર	int, float, void
ફંક્શન નેમ	અનન્ય ઓળખકર્તા	sum, findMax
પેરામીટર્સ	ઇનપુટ ડેટા	(int a, int b)
બોડી	સ્ટેટમેન્ટ સમૂહ	{ return a+b; }

### ઉદાહરણ:

```
\#include <stdio.h>

//  
int sum(int a, int b);

int main() {  
    int num1 = 10, num2 = 20, result;  
  
    //  
    result = sum(num1, num2);  
  
    printf("Sum = %d\n", result);  
  
    return 0;  
}  
  
//  
int sum(int a, int b) {  
    return a + b;  
}
```

### ફક્શન ફ્લો:

```
flowchart LR  
    A["main function"] --> B["sum function"]  
    B --> C["return a + b"]  
    A --> D["a = 10,  
b = 20"]
```

### મેમરી ટ્રીક

“DPCR: Declaration-Parameters-Call-Return”

### પ્રશ્ન 5(c) OR [7 ગુણ]

એરેમાંથી મહત્તમ અને લઘૃતમ element શોધવા માટે C પ્રોગ્રામનો લખો.

### જવાબ

```
\#include <stdio.h>

int main() {  
    int arr[100], n, i;  
    int max, min;  
  
    printf("Enter number of elements: ");  
    scanf("%d", &n);  
  
    printf("Enter %d integers: ", n);  
    for(i = 0; i < n; i++) {  
        scanf("%d", &arr[i]);  
    }  
  
    // max min  
    max = min = arr[0];  
  
    // max min
```

```

for(i = 1; i < n; i++) {
    if(arr[i] > max) {
        max = arr[i];
    }
    if(arr[i] < min) {
        min = arr[i];
    }
}

printf("Maximum element: %d\n", max);
printf("Minimum element: %d\n", min);

return 0;
}

```

#### અણોરિધમ:

1. એરે સાઇઝ અને તત્વો ઇનપુટ કરો
2. max અને min ને પ્રથમ તત્વ સાથે પ્રારંભિક કરો
3. દરેક બાકીના તત્વ માટે:
  - જો તત્વ > max, max અપડેટ કરો
  - જો તત્વ < min, min અપડેટ કરો
4. max અને min ને દર્શાવો

#### ફ્લોચાર્ટ:

```

flowchart LR
    A([Start]) --> B[/Input n/]
    B --> C[/Input array elements/]
    C --> D["max = min = arr[0]"]
    D --> E[i = 1]
    E --> F{i n?}
    F -- Yes --> G["arr[i] max?"]
    G -- Yes --> H["max = arr[i]"]
    G -- No --> I["arr[i] min?"]
    H --> I
    I -- Yes --> J["min = arr[i]"]
    I -- No --> K[i++]
    J --> K
    K --> F
    F -- No --> L[/Display max, min/]
    L --> M([End])

```

#### મેમરી ટ્રીક

``FILLS: First Initialize, Loop through, Look for Small/large''