CSE/IT Batch:Hinglish

Programming in C
Structures and Unions

DPP-01

[NAT]

1.
```
#include <stdio.h>
union u{
  int a;
  char b;
  double d[2];
};
int main()
{
  union u u1;
  printf("%d",(int)sizeof(u1));
  return 0;
}
```
Assume that objects of the type int, char and double occupy 2 bytes, 1 bytes and 4 bytes, respectively.
The memory requirement for variable u1 is _____(in bytes).

[NAT]

2. Consider the following C declaration:
```
struct
{
  long a[3];
  union
 {
   int y;
   float z;
 }u;
} s;
```
Assume that objects of the type int, float and long occupy 2 bytes, 4 bytes and 8 bytes, respectively.
The memory requirement for variable s is _____(in bytes).

[MCQ]

3.
```
#include <stdio.h>
struct s{
   char a, b;
};
void f(struct s *p){
   p->a+=2;
   p->b-=1;
}
int main()
{
   struct s s1, s2, *q;
   s1.a='A'; s1.b='C';
   q=&s1;
   f(q);
   printf("%c\t%c",s1.a, s1.b);
   return 0;
}
```
The output is:
(a) C B
(b) A C
(c) Compilation error
(d) Garbage values

[MCQ]

4.
```
#include <stdio.h>
struct s{
   char a, b;
};
void f(struct s s1){
   s1.a+=3;
   s1.b-=1;
}
int main()
{
   struct s s1;
   s1.a='A'; s1.b='C';
   f(s1);
   printf("%c\t%c",s1.a, s1.b);
   return 0;
}
```

The output is:

(a) C B
(b) A C
(c) Compilation error
(d) Garbage values

[MCQ]

5.
```
#include <stdio.h>
struct s{
   char a, b;
};
void f(struct s s1){
   s1.a+=32;
   s1.b+=32;
}
void g(struct s *p){
   static count=2;
   p->a+=count++;
   p->b+=++count;
}
int main()
{
   struct s s1, s2;
   s1.a='A'; s1.b='C';
   s2.a='B'; s2.b='D';
   f(s1);
   for(int i=0;i<2;i++)g(&s2);
   printf("%c\t%c",s1.a, s1.b);
   printf("\t%c\t%c",s2.a, s2.b);
   return 0;
}
```

The output is:

(a) a c B D
(b) A C B D
(c) A C H N
(d) a c B N

[MCQ]

6.
```
#include <stdio.h>
struct days{
   char *q;
}s[]={"Sunday", "Monday", "Tuesday", "Wednesday",
"Thursday", "Friday", "Saturday"};
int main()
{
  struct days *p=s;
  p=p+3;
  printf("%c", *p++->q);
  printf("%c", *++p->q);
  p=p-2;
  printf("%s",p->q);
  return 0;
}
```

The output string printed is-

(a) WhWednesday
(b) WTTuesday
(c) WTWednesday
(d) WhTuesday

[MSQ]

7. Which of the following statements are INCORRECT?

(a) Functions cannot be defined inside the structure.
(b) Structure variable of the same structure type can be defined inside a structure.
(c) A function may not contain a structure defined in it.
(d) Existing structure cannot be contained in another structure.

[NAT]

8.
```
#include<stdio.h>
#include<string.h>
struct t
{
 char sname[20];
};
int main ()
{
   struct t t1, t2;
   strcpy(t1.sname, "GATEWallah"); //line a
   t2.sname="GATE2023"; //line b
   printf("%s", t1.sname); //line c
   printf("%s", t2.sname); //line d
   return 0;
}
```

The number of lines with error among lines a,b,c,d are _____.

Answer Key

1. **(8)**
2. **(28)**
3. **(a)**
4. **(b)**
5. **(c)**
6. **(d)**
7. **(b, c, d)**
8. **(2)**

Hints and Solutions

1. (8)

The size of the union is equal to the maximum size of the member variables

Here, double d[2] has the maximum size

∴ size of union = (2 × 4) bytes = 8 bytes

2. (28)

The size of the structure variable is the sum of the sizes of all its member variables

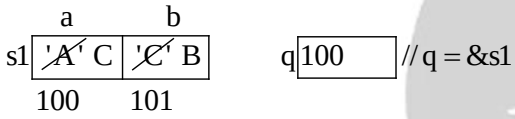
Size of structure = size of long a[3] + size of union

= 8 × 3 + max (4, 4)

= 24 + 4

= 28

3. (a)



f(100)

p | 100

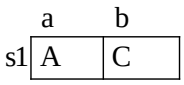
100 → a + = 2; // 100 → a = 'C'

100 → b – = 1; // 100 → b = 'B'

∴ printf() prints 'C B';

4. (b)

main()

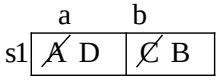


⇒ print("%c\t%c"; s1.a, s1.b);

⇒ A C

f(s1)

entry structure is passed



s1.a + = 3

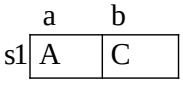
s2.a – = 1

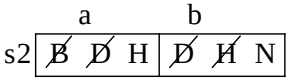
formal parameters are changed.

Actual parameters haven't changed

∴ Output: A C

5. (c)





g(&s2); g(&s2);

f(s1)

entire structure is passed.

⇒ It will change the formal parameters but the actual parameters in main() won't change

g(200)

count | 2 3 4

static

200 → a + = 2; // D

200 → b + = 4; // H

g(200)

count | 4 5 6

200 → a + = 4; // H

200 → b + = 6; // N

∴ Output-

printf("%c\t%c", s1.a, s1.b);

printf("%c\t%c", s2.a, s2.b);

⇒ A C H N

6. (d)

100	101	102	103	104	105	106
S	u	n	d	a	y	\0

200	201	202	203	204	205	206
M	o	n	d	a	y	\0

300	301	302	303	304	305	306	307
T	u	e	s	d	a	y	\0

400	401	402	403	404	405	406	407	409	410
W	e	d	n	e	s	d	a	y	\0

500	501	502	503	504	505	506	507	508
T	h	u	r	s	d	a	y	\0

600	601	602	603	604	605	606
F	r	i	d	a	y	\0

700	701	702	703	704	705	706	707	708
S	a	t	u	r	d	a	y	\0

s	100	200	300	400	500	600	700

↑

array of structures

p	~~800~~	~~812~~	~~816~~	808

printf("%c", *p++ → q);

*812 → 400 ⇒ w

printf("%c", *++p → q);

*++812 → 500 ⇒ *++500

⇒ *++501

⇒ h

p=p–2;//p=808

printf("%s", p → q);

808 → 300 ⇒ Tuesday

∴ Output: WhTuesday

7. (b, c, d)

(a) CORRECT. Functions cannot be defined inside the structure

(b) INCORRECT. Structure variable of the same structure type cannot be defined inside structure.

(c) INCORRECT. A function can contain a structure defined in it.

(d) INCORRECT. Existing structure can be contained in another structure

8. (2)

line b: ERROR. Constant base address cannot be changed

line d: ERROR. As line 2 has error, line 4 cannot be executed.