|  |
| --- |
|  |

**Predict the output or error(s) for the following:**  
**1.     struct aaa{  
struct aaa \*prev;  
int i;  
struct aaa \*next;  
};  
main()**{ **struct aaa abc,def,ghi,jkl;  
 int x=100;** **abc.i=0;abc.prev=&jkl;** **abc.next=&def;** **def.i=1;def.prev=&abc;def.next=&ghi;** **ghi.i=2;ghi.prev=&def;** **ghi.next=&jkl;** **jkl.i=3;jkl.prev=&ghi;jkl.next=&abc;  
 x=abc.next->next->prev->next->i;  
 printf("%d",x);**}  
  
Answer:  
2

Explanation:  
above all statements form a double circular linked list;  
abc.next->next->prev->next->i   
this one points to "ghi" node the value of at particular node is 2.

**2.       struct point  
 {  
 int x;  
 int y;  
 };  
struct point origin,\*pp;  
main()**{ **pp=&origin;  
printf("origin is(%d%d)\n",(\*pp).x,(\*pp).y);  
printf("origin is (%d%d)\n",pp->x,pp->y);  
}**  
              
Answer:  
origin is(0,0)  
origin is(0,0)   
  
Explanation:  
pp is a pointer to structure. we can access the elements of the structure either with arrow mark or with indirection operator.   
Note:   
Since structure point  is globally declared x & y are initialized as zeroes

**3.       main()**{ **int i=\_l\_abc(10);  
             printf("%d\n",--i);**} **int \_l\_abc(int i)**{ **return(i++);**}  
  
Answer:  
9  
  
Explanation:   
return(i++) it will first return i and then increments. i.e. 10 will be returned.

**4.       main()**{ **char \*p;  
 int \*q;  
 long \*r;  
 p=q=r=0;  
 p++;  
 q++;  
 r++;  
 printf("%p...%p...%p",p,q,r);**}  
  
Answer:  
0001...0002...0004  
  
Explanation:  
++ operator  when applied to pointers increments address according to their corresponding data-types.

**5.      main()**{ **char c=' ',x,convert(z);  
 getc(c);  
 if((c>='a') && (c<='z'))  
 x=convert(c);  
 printf("%c",x);**} **convert(z)**{ **return z-32;**}  
  
Answer:   
Compiler error  
  
Explanation:  
declaration of convert and format of getc() are wrong.

**6.      main(int argc, char \*\*argv)**{ **printf("enter the character");  
 getchar();  
 sum(argv[1],argv[2]);**} **sum(num1,num2)  
int num1,num2;**{ **return num1+num2;**}  
  
Answer:  
Compiler error.  
  
Explanation:  
argv[1] & argv[2] are strings. They are passed to the function sum without converting it to integer values.

**7.       # include   
int one\_d[]={1,2,3};  
main()**{ **int \*ptr;   
 ptr=one\_d;  
 ptr+=3;  
 printf("%d",\*ptr);**}  
  
Answer:  
garbage value  
  
Explanation:  
ptr pointer is pointing to out of the array range of one\_d.

**Predict the output or error(s) for the following:  
1.    void main()**{ **int  const \* p=5;  
            printf("%d",++(\*p));**}  
  
Answer:  
Compiler error: Cannot modify a constant value.   
Explanation:      
p is a pointer to a "constant integer". But we tried to change the value of the "constant integer".

**2.    main()**{ **char s[ ]="man";  
            int i;  
            for(i=0;s[ i ];i++)  
            printf("\n%c%c%c%c",s[ i ],\*(s+i),\*(i+s),i[s]);**}  
  
Answer:  
                        mmmm  
                       aaaa  
                       nnnn  
Explanation:  
s[i], \*(i+s), \*(s+i), i[s] are all different ways of expressing the same idea. Generally  array name is the base address for that array. Here s is the base address. i is the index number/displacement from the base address. So, indirecting it with \* is same as s[i]. i[s] may be surprising. But in the  case of  C  it is same as s[i].

**3.      main()**{ **float me = 1.1;  
            double you = 1.1;  
            if(me==you)  
printf("I love U");  
else  
                        printf("I hate U");**}  
  
Answer:   
I hate U  
  
Explanation:  
For floating point numbers (float, double, long double) the values cannot be predicted exactly. Depending on the number of bytes, the precession with of the value  represented varies. Float takes 4 bytes and long double takes 10 bytes. So float stores 0.9 with less precision than long double.  
Rule of Thumb:   
Never compare or at-least be cautious when using floating point numbers with relational operators (== , >, <, <=, >=,!= ) .

**4.      main()  
            {  
            static int var = 5;  
            printf("%d ",var--);  
            if(var)  
                        main();  
            }**  
Answer:  
5 4 3 2 1  
  
Explanation:  
When static storage class is given, it is initialized once. The change in the value of a static variable is retained even between the function calls. Main is also treated like any other ordinary function, which can be called recursively.

**5.      main()**{ **int c[ ]={2.8,3.4,4,6.7,5};  
             int j,\*p=c,\*q=c;  
             for(j=0;j<5;j++) {  
                        printf(" %d ",\*c);  
                        ++q;     }  
             for(j=0;j<5;j++){  
printf(" %d ",\*p);  
++p;     }**}  
   
Answer:  
                        2 2 2 2 2 2 3 4 6 5  
  
Explanation:   
Initially pointer c is assigned to both p and q. In the first loop, since only q is incremented and not c , the value 2 will be printed 5 times. In second loop p itself is incremented. So the values 2 3 4 6 5 will be printed.

**6.      main()**{ **extern int i;  
            i=20;  
printf("%d",i);**}  
   
Answer:    
Linker Error : Undefined symbol '\_i'  
Explanation:   
                        extern storage class in the following declaration,  
                                    extern int i;  
specifies to the compiler that the memory for i is allocated in some other program and that address will be given to the current program at the time of linking. But linker finds that no other variable of name i is available in any other program with memory space allocated for it. Hence a linker error has occurred .

|  |  |
| --- | --- |
| [techpreparation-homepage](http://techpreparation.com/) |  |

|  |  |
| --- | --- |
| |  | | --- | | [**Home**](http://techpreparation.com/)  [**Interview Questions**](http://techpreparation.com/interview-questions.htm)  [**Aptitude Questions**](http://techpreparation.com/aptitude-questions.htm)  [**Tutorials**](http://techpreparation.com/tutorials.htm)  [**Placement Papers**](http://techpreparation.com/placement-papers.htm)  [**Search**](http://techpreparation.com/search.htm)  [**Resume Guide**](http://techpreparation.com/resume-guide.htm)  [**Soft Skills**](http://techpreparation.com/soft-skills.htm)  [**Video**](http://techpreparation.com/video.htm)  [**Forum**](http://techpreparation.com/post/)  [**Blog**](http://techpreparation.com/blog/) | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Technical Interview Questions [C++ Interview Questions](http://techpreparation.com/cpp-interview-question-answer1.htm) [C# Interview Questions](http://techpreparation.com/c-sharp-interview-questions-answers1.htm) [C Interview Questions](http://techpreparation.com/c-interview-questions-answers1.htm)                               [.........More](http://techpreparation.com/interview-questions.htm)  Aptitude Interview Questions [C ++ Aptitude Questions](http://techpreparation.com/aptitute-questions/c-cpp-aptitude-questions1.htm) [C Aptitude Questions 1](http://techpreparation.com/aptitute-questions/c-aptitude-questions1.htm)                               [.........More](http://techpreparation.com/aptitude-questions.htm)  Tutorials [C Tutorial](http://techpreparation.com/tutorials/c-programming-tutorials.htm) [C++ Tutorial](http://techpreparation.com/tutorials/c++-programming-tutorials.htm)                               [.........More](http://techpreparation.com/tutorials.htm)  Programming Source Codes [C/C++ Source Codes](http://techpreparation.com/source-codes/c-source-code.htm) [C# Source Codes](http://techpreparation.com/source-codes/c-sharp-source-code.htm)                               [.........More](http://techpreparation.com/source-codes.htm)  Soft Skills [Communication Skills](http://techpreparation.com/soft-skills/communication.htm) [Leadership Skills](http://techpreparation.com/soft-skills/leadership-skills.htm)                               [.........More](http://techpreparation.com/soft-skills.htm)  Top of Form   |  |  | | --- | --- | | Subscribe to our Newsletters | | | Name: |  | | Email: |  | |  | |   Bottom of Form  [http://s9.addthis.com/button1-bm.gif](http://www.addthis.com/bookmark.php) |  | Top of Form   |  | | --- | |  |   Bottom of Form  **C Aptitude Questions and Answers**   |  | | --- | |  |   **Predict the output or error(s) for the following:** **1.     struct aaa{ struct aaa \*prev; int i; struct aaa \*next; }; main()** { **struct aaa abc,def,ghi,jkl;  int x=100;**  **abc.i=0;abc.prev=&jkl;**  **abc.next=&def;**  **def.i=1;def.prev=&abc;def.next=&ghi;**  **ghi.i=2;ghi.prev=&def;**  **ghi.next=&jkl;**  **jkl.i=3;jkl.prev=&ghi;jkl.next=&abc;  x=abc.next->next->prev->next->i;  printf("%d",x);** }  Answer: 2  Explanation: above all statements form a double circular linked list; abc.next->next->prev->next->i  this one points to "ghi" node the value of at particular node is 2.  **2.       struct point  {  int x;  int y;  }; struct point origin,\*pp; main()** { **pp=&origin; printf("origin is(%d%d)\n",(\*pp).x,(\*pp).y); printf("origin is (%d%d)\n",pp->x,pp->y); }**              Answer: origin is(0,0) origin is(0,0)   Explanation: pp is a pointer to structure. we can access the elements of the structure either with arrow mark or with indirection operator.  Note:  Since structure point  is globally declared x & y are initialized as zeroes  **3.       main()** { **int i=\_l\_abc(10);              printf("%d\n",--i);** } **int \_l\_abc(int i)** { **return(i++);** }  Answer: 9  Explanation:  return(i++) it will first return i and then increments. i.e. 10 will be returned.  **4.       main()** { **char \*p;  int \*q;  long \*r;  p=q=r=0;  p++;  q++;  r++;  printf("%p...%p...%p",p,q,r);** }  Answer: 0001...0002...0004  Explanation: ++ operator  when applied to pointers increments address according to their corresponding data-types.  **5.      main()** { **char c=' ',x,convert(z);  getc(c);  if((c>='a') && (c<='z'))  x=convert(c);  printf("%c",x);** } **convert(z)** { **return z-32;** }  Answer:  Compiler error  Explanation: declaration of convert and format of getc() are wrong.  **6.      main(int argc, char \*\*argv)** { **printf("enter the character");  getchar();  sum(argv[1],argv[2]);** } **sum(num1,num2) int num1,num2;** { **return num1+num2;** }  Answer: Compiler error.  Explanation: argv[1] & argv[2] are strings. They are passed to the function sum without converting it to integer values.  **7.       # include  int one\_d[]={1,2,3}; main()** { **int \*ptr;   ptr=one\_d;  ptr+=3;  printf("%d",\*ptr);** }  Answer: garbage value  Explanation: ptr pointer is pointing to out of the array range of one\_d.   Page Numbers :   1      [2](http://techpreparation.com/aptitute-questions/c-aptitude-question2.htm)      [3](http://techpreparation.com/aptitute-questions/c-aptitude-question3.htm)      [4](http://techpreparation.com/aptitute-questions/c-aptitude-question4.htm)      [5](http://techpreparation.com/aptitute-questions/c-aptitude-question5.htm)      [6](http://techpreparation.com/aptitute-questions/c-aptitude-question6.htm)      [7](http://techpreparation.com/aptitute-questions/c-aptitude-question7.htm)      [8](http://techpreparation.com/aptitute-questions/c-aptitude-question8.htm)      [9](http://techpreparation.com/aptitute-questions/c-aptitude-question9.htm)      [10](http://techpreparation.com/aptitute-questions/c-aptitude-question10.htm)      [11](http://techpreparation.com/aptitute-questions/c-aptitude-question11.htm)      [12](http://techpreparation.com/aptitute-questions/c-aptitude-question12.htm)  [http://s9.addthis.com/button1-bm.gif](http://www.addthis.com/bookmark.php)  Have a Question ? [post your questions](http://techpreparation.com/post/) here. It will be answered as soon as possible.  Check [C Aptitude Questions](http://techpreparation.com/aptitute-questions/c-aptitude-questions1.htm) for more C Aptitude Interview Questions with Answers  Check [Placement Papers](http://techpreparation.com/placement-papers.htm) for more IT Companies Paper |

vv