**Enumerations and Typedefs in C**

1. A user defined data type, which is used to assign names to integral constants is called \_\_\_\_\_\_\_\_\_\_\_\_  
a) Union  
b) Array  
c) Structure  
d) Enum

2. What will be the output of the following C code?

advertisement

#include<stdio.h>

**enum** colour

{

blue, red, yellow

};

main()

{

**enum** colour c;

c=yellow;

printf("%d",c);

}

a) 1

b) 2  
c) 0  
d) error

3. Point out the error (if any) in the following C code?

#include<stdio.h>

**enum** hello

{

a,b,c;

};

main()

{

**enum** hello m;

printf("%d",m);

}

a) No error  
b) Error in the statement: a,b,c;  
c) Error in the statement: enum hello m;  
d) Error in the statement: printf(“%d”,m);

4. String handling functions such as strcmp(), strcpy() etc can be used with enumerated types.  
a) True  
b) False

5. What will be the output of the following C code?

#include<stdio.h>

**enum** hello

{

a,b=99,c,d=-1

};

main()

{

**enum** hello m;

printf("%d**\n**%d**\n**%d**\n**%d**\n**",a,b,c,d);

}

a)

1

99

100

-1

b) Error  
c)

0

99

100

-1

d)

0

1

2

3

6. Pick the incorrect statement with respect to enums.  
a) Two enum symbols cannot have the same value  
b) Only integer constants are allowed in enums  
c) It is not possible to change the value of enum symbols  
d) Enum variables are automatically assigned values if no value is specified

7. What will be the output of the following C code?

#include<stdio.h>

**enum** sanfoundry

{

a=2,b=3.56

};

**enum** sanfoundry s;

main()

{

printf("%d%d",a,b);

}

a) 2 3  
b) 0 1  
c) 2 3.56  
d) Error

8. What will be the output of the following C code?

#include<stdio.h>

**enum** class

{

a,b,c

};

**enum** class m;

main()

{

printf("%d",sizeof(m));

}

a) 3  
b) Same as the size of an integer  
c) 3 times the size of an integer  
d) Error

9. What will be the output of the following C code?

#include<stdio.h>

**enum** hi{a,b,c};

**enum** hello{c,d,e};

main()

{

**enum** hi h;

h=b;

printf("%d",h);

return 0;

}

a) 2  
b) 1  
c) Error  
d) 0

10. What will be the output of the following C code?

#include<stdio.h>

**enum** sanfoundry

{

a,b,c=5

};

**enum** sanfoundry s;

main()

{

c++;

printf("%d",c);

}

a) Error  
b) 5  
c) 6  
d) 2

1. What will be the output of the following C code?

main()

{

**enum** resut {pass, fail};

**enum** result s1,s2;

s1=pass;

s2=fail;

printf("%d",s1);

}

a) error  
b) pass  
c) fail  
d) 0

2. What will be the output of the following C code?

#include <stdio.h>

**enum** example {a = 1, b, c};

**enum** example example1 = 2;

**enum** example answer()

{

return example1;

}

int main()

{

(answer() == a)? printf("yes"): printf("no");

return 0;

}

a) yes  
b) no  
c) 2  
d) error

3. What will be the output of the following C code?

#include<stdio.h>

#define MAX 4

**enum** sanfoundry

{

a,b=3,c

};

main()

{

if(MAX!=c)

printtf("hello");

else

printf("welcome");

}

a) error  
b) hello  
c) welcome  
d) 2

4. Arithmetic operations such as addition, subtraction, multiplication and division are allowed on enumerated constants.  
a) True  
b) False

5. Point out the error( if any) in the following code.

#include<stdio.h>

**enum** sanfoundry

{

a,b,c

};

**enum** sanfoundry g;

main()

{

g++;

printf("%d",g);

}

a) Error in the statement: a,b,c  
b) Error in the statement: enum sanfoundry g;  
c) Error in the statement: g++  
d) No error

6. What will be the output of the following C code if input given is 2?

#include<stdio.h>

**enum** day

{

a,b,c=5,d,e

};

main()

{

printf("Enter the value for a");

scanf("%d",a);

printf("%d",a);

}

a) 2  
b) 0  
c) 3  
d) Error

7. What will be the output of the following C code if the code is executed on a 32 bit platform?

#include <stdio.h>

**enum** sanfoundry

{

c = 0,

d = 10,

h = 20,

s = 3

} a;

int main()

{

a = c;

printf("Size of enum variable = %d bytes", sizeof(a));

return 0;

}

a) Error  
b) Size of enum variable = 2 bytes  
c) Size of enum variable = 4 bytes  
d) Size of enum variables = 8 bytes

8. What will be the output of the following C code?

#include<stdio.h>

**enum** sanfoundry

{

a=1,b,c,d,e

};

int main()

{

printf("%d",b\*c+e-d);

}

a) Error  
b) 7  
c) 2  
d) 4

9. What will be the output of the following C code?

#include<stdio.h>

**enum** sanfoundry

{

a,b,c=5

};

int main()

{

**enum** sanfoundry s;

b=10;

printf("%d",b);

}

a) Error  
b) 10  
c) 1  
d) 4

10. What will be the output of the following C code?

#include<stdio.h>

**enum** sanfoundry

{

a=1,b

};

**enum** sanfoundry1

{

c,d

};

int main()

{

**enum** sanfoundry1 s1=c;

**enum** sanfoundry1 s=a;

**enum** sanfoundry s2=d;

printf("%d",s);

printf("%d",s1);

printf("%d",s2);

}

a) Error  
b) 011  
c) 110  
d) 101

**TYPEDEF :**

1. Which of the following keywords is used to define an alternate name for an already existing data type?  
a) default  
b) volatile  
c) typedef  
d) static

2. We want to create an alias name for an identifier of the type unsigned long. The alias name is: ul. The correct way to do this using the keyword typedef is \_\_\_\_\_\_\_\_\_\_\_\_  
a) typedef unsigned long ul;  
b) unsigned long typedef ul;  
c) typedef ul unsigned long;  
d) ul typedef unsigned long;

3. What will be the output of the following C code?

#include<stdio.h>

main()

{

typedef int a;

a b=2, c=8, d;

d=(b\*2)/2+8;

printf("%d",d);

}

a) 10  
b) 16  
c) 8  
d) error

4. WWhat will be the output of the following C code? (If the name entered is: Sanfoundry)

#include<stdio.h>

#include<string.h>

typedef struct employee

{

char name[50];

int salary;

} e1;

void main( )

{

printf("Enter Employee name");

scanf("%s",e1.name);

printf("**\n**%s",e1.name);

}

a) Sanfoundry.name  
b) nSanfoundry  
c) Sanfoundry  
d) Error

5. The keyword typedef cannot be used to give alias names to pointers.  
a) True  
b) False

6. What is the size of myArray in the code shown below? (Assume that 1 character occupies 1 byte)

typedef char x[10];

x myArray[5];

a) 5 bytes  
b) 10 bytes  
c) 40 bytes  
d) 50 bytes

7. We want to declare x, y and z as pointers of type int. The alias name given is: intpt The correct way to do this using the keyword typedef is:  
a)

int typedef\* intptr;

int x,y,z;

b)

typedef\* intptr;

int x,y,z;

c)

int typedef\* intptr;

intptr x,y,z;

d)

typedef int\* intptr;

intptr x,y,z; 

8. Consider this statement: typedef enum good {a, b, c} hello; Which of the following statements is incorrect about hello?  
a) hello is a typedef of enum good  
b) hello is a structure  
c) hello is a variable of type enum good  
d) the statement shown above is erroneous

9. One of the major difference between typedef and #define is that typedef interpretation is performed by the \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ whereas #define interpretation is performed by the \_\_\_\_\_\_\_\_\_\_\_\_\_  
a) pre-processor, compiler  
b) user, pre-processor  
c) compiler, pre-processor  
d) compiler, user

10. What will be the output of the following C code?

#include<stdio.h>

int main()

{

typedef union a

{

int i;

char ch[2];

}hello;

hello u;

u.ch[0] = 3;

u.ch[1] = 2;

printf("%d, %d", u.ch[0], u.ch[1]);

return 0;

}

a) 2, 3  
b) 3, 2  
c) 32  
d) error

**C MCQs on String Operations**

**String Operations – 1:**

1. There are two groups of string functions defined in the header <string.h>. What are they?  
a) first group names beginning with str; second group names beginning with mem  
b) first group names beginning with str; second group names beginning with is  
c) first group names beginning with string; second group names beginning with mem  
d) first group names beginning with str; second group names beginning with type

2. What is the use of function char \*strchr(ch, c)?  
a) return pointer to first occurrence of ch in c or NULL if not present  
b) return pointer to first occurrence of c in ch or NULL if not present  
c) return pointer to first occurrence of ch in c or ignores if not present  
d) return pointer to first occurrence of cin ch or ignores if not present

3. Which code from the given option return pointer to last occurrence of c in ch or NULL if not present?  
a) char \*strchr(ch, c)  
b) char \*strrchr(ch, c)  
c) char \*strncat(ch, c)  
d) char \*strcat(ch, c)

4. Which among the given options compares atmost n characters of string ch to string s?  
a) int strncmp(ch, s, n)  
b) int strcmp(ch, s)  
c) int strncmp(s, ch, n)  
d) int strcmp(s, ch)

5. Which among the given options is the right explanation for the statement size\_t strcspn(c, s)?  
a) return length of prefix of s consisting of characters not in c  
b) return length of prefix of s consisting of characters present in c  
c) return length of prefix of c consisting of characters not in s  
d) return length of prefix of c consisting of characters present in s

6. The mem functions are meant for \_\_\_\_\_\_\_  
a) returning a pointer to the token  
b) manipulating objects as character arrays  
c) returning a pointer for implemented-defined string  
d) returning a pointer to first occurrence of string in another string

7. What is the function of void \*memset(s, c, n)?  
a) places character s into first n characters of c, return c  
b) places character c into first n characters of s, return s  
c) places character s into first n characters of c, return s  
d) places character c into first n character of s, return c

8. Functions whose names begin with “strn”  
a) manipulates sequences of arbitrary characters  
b) manipulates null-terminated sequences of characters  
c) manipulates sequence of non – null characters.  
d) returns a pointer to the token

9. Which of the following is the right syntax to copy n characters from the object pointed to by s2 into the object pointed to by s1?  
a) void \*memcpy(void \*s1,const void \*s2,size\_t n);  
b) void \*memcpy(void \*s2, const void \*s1, size\_t n);  
c) void memcpy(void \*s1,const void \*s2, size\_t n);  
d) void memcpy(void \*s2,const void \*s1,size\_t n);

10. What does the following function returns void \*memmove(void \*s1,const void s2, size\_t n);?  
a) returns the value of s1  
b) returns the value of s2  
c) doesn’t return any value  
d) returns the value of s1 and s2

**String Operations – 2:**

1. Which among the following is Copying function?  
a) memcpy()  
b) strcopy()  
c) memcopy()  
d) strxcpy()

2. Which function will you choose to join two words?  
a) strcpy()  
b) strcat()  
c) strncon()  
d) memcon()

advertisement

3. The \_\_\_\_\_\_ function appends not more than n characters.  
a) strcat()  
b) strcon()  
c) strncat()  
d) memcat()

4. What will strcmp() function do?  
a) compares the first n characters of the object  
b) compares the string  
c) undefined function  
d) copies the string

5. What is the prototype of strcoll() function?  
a) int strcoll(const char \*s1,const char \*s2)  
b) int strcoll(const char \*s1)  
c) int strcoll(const \*s1,const \*s2)  
d) int strcoll(const \*s1)

6. What is the function of strcoll()?  
a) compares the string, result is dependent on the LC\_COLLATE  
b) copies the string, result is dependent on the LC\_COLLATE  
c) compares the string, result is not dependent on the LC\_COLLATE  
d) copies the string, result is not dependent on the LC\_COLLATE

7. Which of the following is the variable type defined in header string. h?  
a) sizet  
b) size  
c) size\_t  
d) size-t

8. NULL is the macro defined in the header string. h.  
a) true  
b) false

9. What will be the output of the following C code?

const char pla[] = "string1";

const char src[] = "string2";

printf("Before memmove place= %s, src = %s**\n**", pla, src);

memmove(pla, src, 7);

printf("After memmove place = %s, src = %s**\n**", pla, src);

a) Before memmove place= string1, src = string2 After memmove place = string2, src = string2  
b) Before memmove place = string2, src = string2 After memmove place= string1, src = string2  
c) Before memmove place = string2, src = string1 After memmove place= string2, src =string2  
d) Before memmove place= string1, src = string2 After memmove place=string1, src = string1

10. What will be the output of the following C code?

const char str1[]="ABCDEF1234567";

const char str2[] = "269";

len = strcspn(str1, str2);

printf("First matching character is at %d**\n**", len + 1);

a) First matching character is at 8  
b) First matching character is at 7  
c) First matching character is at 9  
d) First matching character is at 12

**String Operations – 3:**

1. What is the return value of strxfrm()?  
a) length of the transformed string, not including the terminating null-character  
b) length of the transformed string, including the terminating null-character  
c) display the transformed string, not including the terminating null character  
d) display the transformed string, not including the terminating null-character

2. Is there any function declared as strstr()?  
a) true  
b) false

3. The C library function \_\_\_\_\_\_\_\_\_ breaks string s1 into a series of tokens using the delimiter s2.  
a) char \*strtok(char \*s1,const char \*s2);  
b) char \*strtok(char \*s2,const char \*s1);  
c) char \*strstr(char \*s1,const char \*s2);  
d) char \*strstr(char \*s2,const char \*s1);

4. The\_\_\_\_\_\_function returns a pointer to the first character of a token.  
a) strstr()  
b) strcpy()  
c) strspn()  
d) strtok()

5. which of the following function returns a pointer to the located string or a null pointer if string is not found.  
a) strtok()  
b) strstr()  
c) strspn()  
d) strrchr()

6. Which of the given function is used to return a pointer to the located character?  
a) strrchr()  
b) strxfrm()  
c) memchar()  
d) strchar()

7. The strpbrk() function is used to return a pointer to the character, or a null pointer if no character from s2 occurs in s1.  
a) true  
b) false

8. What will be the output of the following C code?

const char str1[] = "abcdef";

const char str2[] = "fgha";

char \*mat;

mat= strpbrk(str1, str2);

if(mat)

printf("First matching character: %c**\n**", \*mat);

else

printf("Character not found");

a) g  
b) a  
c) h  
d) f

9. What will be the output of the following C code?

char str1[] = "Helloworld ";

char str2[] = "Hello";

len = strspn(str1, str2);

printf("Length of initial segment matching %d**\n**", len );

a) 6  
b) 5  
c) 4  
d) no match

10. The\_\_\_\_\_\_ function returns the number of characters that are present before the terminating null character.  
a) strlength()  
b) strlen()  
c) strlent()  
d) strchr()

**String Operations – 4:**

1. What will be returned in the following C code?

size- t strlen(const char \*s)

const char \*sc;

for(sc = s; \*sc!= ' \ O ' ; ++sc)

return(sc - s) ;

a) number of characters equal in sc  
b) length of string s  
c) doesn’t return any value  
d) displays string

2. The function strsp() is the complement of strcspn().  
a) true  
b) false

3. What will the following C code do?

char \* strrchr(const char \*s, int c )

char ch = c;

char \*sc;

for(sc = NULL; ; ++s)

if(\*s == ch)

SC = 9;

i f (\*s =='\O' )

return (( char \*) s);

a) find last occurrence of c in char s[ ].  
b) find first occurrence of c in char s[ ].  
c) find the current location of c in char s[ ].  
d) There is error in the given code

4. This function offers the quickest way to determine whether two character sequences of the same known length match character for the character up to and including any null character in both.  
a) strcmp()  
b) memcmp()  
c) strncmp()  
d) no such function

5. What will be the output of the following C code?

char str1[15];

char str2[15];

int mat;

strcpy(str1, "abcdef");

strcpy(str2, "ABCDEF");

mat= strncmp(str1, str2, 4);

if(mat< 0)

printf("str1 is not greater than str2");

else if(mat> 0)

printf("str2 is is not greater than str1");

else

printf("both are equal");

a) str1 is not greater than str2  
b) str2 is not greater than str1  
c) both are equal  
d) error in given code

6. What will be the output of the following C code?

void \*memset(void \*c, int c, size-t n)

unsigned char ch = c;

unsigned char \*su;

for (su = s; 0 < n; ++su, --n)

<br>

\*su = ch;

<br>

a) store c throughout unsigned char s[n]  
b) store c throughout signed char s[n]  
c) find first occurrence of c in s[n]  
d) find last occurrence of c in s[n]

7. Use\_\_\_\_\_\_\_to determine the null-terminated message string that corresponds to the error code errcode.  
a) strerror()  
b) strstr()  
c) strxfrm()  
d) memset()

8. What will be the output of the following C code?

const char str1[10]="Helloworld";

const char str2[10] = "world";

char \*mat;

mat = strstr(str1, str2);

printf("The substring is:%s\n", mat);

a) The substring is:world  
b) The substring is:Hello  
c) The substring is:Helloworld  
d) error in the code

9. void \*memcpy(void \*dest, const void \*src, size\_t n) What does the following code do?  
a) copies n characters from src to dest  
b) copies n character from dest to src  
c) transform n character from dest to src  
d) transform n character from src to dest

10. What will the given C code do?

int memcmp(const void \*str1, const void \*str2, size\_t n)

a) compares the first n bytes of str1 and str2  
b) copies the first n bytes of str1 to str2  
c) copies the first n bytes of str2 to str1  
d) invalid function

**Character Handling – 1:**

1. Which header declares several functions useful for testing and mapping characters?  
a) assert.h  
b) stdio.h  
c) ctype.h  
d) errno.h

2. The\_\_\_\_\_\_function tests for any character for which isalpha or isdigit is true.  
a) isxdigit()  
b) isspace()  
c) isalnum()  
d) isupper()

3. What does the following C code do?

int iscntrl( int c);

a) checks if character is upper case  
b) checks if character is lower case  
c) tests for any control character  
d) no function as such

4. What do the following C function do?

int isgraph(int c);

a) tests for only space character  
b) tests for only digit  
c) tests for only lower case  
d) tests for any printing character

5. The isdigit function tests for any decimal-digit character.  
a) true  
b) false

6. Which function returns true only for the characters defined as lowercase letters?  
a) islow()  
b) islower()  
c) isalpa()  
d) isalnum()

7. This function checks whether the passed character is white-space.  
a) ispunct()  
b) isgraph()  
c) isspace()  
d) isalpha()

8. The standard white-space characters are the following: space (' '), form feed (' \ f '),  
newline (' \n') , horizontal tab (' \tr), and vertlcal tab (' \v') can be tested with function.  
a) ispunct()  
b) isalpha()  
c) isgraph()  
d) isspace()

9. Which function tests for any character that is an uppercase letter.  
a) iscntrl()  
b) ispunct()  
c) isdigit()  
d) isupper()

10. The\_\_\_\_\_\_function tests for any hexadecimal-digit character.  
a) iscntrl()  
b) ispunct()  
c) isgraph()  
d) isxdigit()

**Character Handling – 2:**

1. The\_\_\_\_\_\_\_\_function converts an uppercase letter to the corresponding lowercase letter.  
a) islower()  
b) isupper()  
c) toupper()  
d) tolower()

2. The toupper() function converts a \_\_\_\_\_\_ to the corresponding \_\_\_\_\_\_  
a) uppercase, lowercase  
b) lowercase, uppercase  
c) binary, decimal  
d) decimal, binary

3. fgetc, getc, getchar are all declared in \_\_\_\_\_\_\_\_  
a) stdio. h  
b) ctype. h  
c) assert. h  
d) stdarg. h

4. isalpha() function is used to detect characters both in uppercase and lowercase.  
a) true  
b) false

5. What will be the output of the following C code?

int ch= ' ';

if(isgraph(ch))

printf("ch = %c can be printed **\n**",ch);

else

printf("ch=%c cannot be printed **\n**",ch);

a) ch = ‘ ‘ can be printed  
b) ch = ‘ ‘ cannot be printed  
c) compile error  
d) run-time error

6. The C library function checks whether the passed character is printable.  
a) isgraph()  
b) isalpha()  
c) isprint()  
d) ispunct()

7. What will be the output of the following C code?

char ch[ ] = "0xC";

if(isxdigit(ch[ ]))

printf("ch = %s is hexadecimal character **\n**",ch);

else

printf("ch = %s is not hexadecimal character **\n**",ch);

a) ch = 0xC is hexadecimal character  
b) ch = 0xC is not hexadecimal character  
c) compile error  
d) run-time error

8. Which among the following option is the full set of character class Hexadecimal digits?  
a) { 0 1 2 3 4 5 6 7 8 9 A B C D E F }  
b) { 0 1 2 3 4 5 6 7 8 9 a b c d e f }  
c) { 0 1 2 3 4 5 6 7 8 9 A B C D E F a b c d e f }  
d) { 0 1 2 3 4 5 6 7 8 9}

9. What will be the output of the following C code?

int i = 0;

char c;

char str[ ] = "Little Star";

while(str[i])

{

putchar (toupper(str[i]));

i++;

}

a) little star  
b) lITTLE sTAR  
c) LITTLE STAR  
d) Little Star

10. What will be the output of the following C code?

int ch = '**\t**';

if(isprint(ch))

printf("ch = |%c| printable **\n**", ch);

else

printf("ch= |%c| not printable **\n**",ch);

a) ch = |\t| printable  
b) ch = |\t| not printable  
c) ch = | | printable  
d) ch = | | not printable

**Error Handling:**

1. \_\_\_\_\_\_\_ occurs when a result is too large in magnitude to represent errors as a floating-point value of the required type.  
a) underflow  
b) signiﬁcance loss  
c) domain  
d) overflow

2. What occurs when a result has nowhere near the number of signiﬁcant digits indicated by its type.  
a) domain  
b) underflow  
c) overflow  
d) signiﬁcance loss

3. What error occurs when a result is undeﬁned for a given argument value?  
a) signiﬁcance loss  
b) underflow  
c) overflow  
d) domain

4.\_\_\_\_\_\_ is reported on a domain error.  
a) EDOM  
b) ERANGE  
c) Signiﬁcance loss  
d) Underflow

5. ERANGE is reported on an overflow or an underflow.  
a) true  
b) false

6. What will be the output of the following C code?

errno = 0;

y = sqrt(2);

if(errno == EDOM)

printf(&quot;Invalid value\n&quot;);

else

printf(&quot;Valid value\n&quot;);

a) Invalid value  
b) Valid value  
c) No output  
d) Compile error

7. What will be the output of the following C code?

errno = 0;

y = sqrt(-10);

if(errno == EDOM)

printf(&quot;Invalid value \n&quot;);

else

printf(&quot;Valid value\n&quot;);

a) Invalid value  
b) Valid value  
c) No output  
d) Compile error

8. errno causes trouble in two subtler ways(vague and explicit).  
a) true  
b) false  
View Answer

9. No library function will store a zero in errno.  
a) true  
b) false

10. \_\_\_\_\_\_\_\_\_\_ tells the compiler that this data is deﬁned somewhere and will be connected with the linker.  
a) errno  
b) extern  
c) variable  
d) yvals

**Multiple Choice Questions on C Library**

**Mathematical Functions – 1:**

1. Which of the following header declares mathematical functions and macros?  
a) math.h  
b) assert.h  
c) stdmat. h  
d) stdio. h

2. All the functions in this library take as a parameter and return as the output.  
a) double, int  
b) double, double  
c) int, double  
d) int, int

3. HUGE\_VAL macro is used when the output of the function may not be \_\_\_\_\_\_\_\_\_\_\_  
a) floating point numbers  
b) integer number  
c) short int  
d) long int

4. What error occurs if an input argument is outside the domain over which the mathematical function is deﬁned?  
a) domain error  
b) range error  
c) no error  
d) domain and range error

5. A range error occurs if the result of the function cannot be represented as a value.  
a) int  
b) short int  
c) double  
d) float

6. If the result overflows, the function returns the value of the macro HUGE\_VAL, carrying the same sign except for the \_\_\_\_\_\_\_\_\_ function as the correct value of the function.  
a) sin  
b) cos  
c) cosec  
d) tan

7. If the result underflow, the function returns zero.  
a) true  
b) false

8. For the given math function, an error occurs if the arguments are not in the range [-1, +1].

double acos(double x);

a) range error  
b) domain error  
c) no error  
d) domain and range error

9. Which function returns the arc sine in the range [-pi/2, +pi/2] radians?  
a) arcsin()  
b) asin()  
c) sin()  
d) asine()

10. What will be the output of the following C code?

double x, deg, rad;

x = 1.0;

val = 180.0 / 3.14;

deg = atan (x) \* val;

printf("The arc tangent of %lf is %lf degrees", x, deg);

a) The arc tangent of 1.000000 is 45.000000 degrees  
b) The arc tangent of 1.000 is 45.000degrees  
c) The arc tangent of 1 is 45 degrees  
d) The arc tangent of 1.0000is 45.0000degrees

**Mathematical Functions – 2:**

1. What does the given C code do?

double atan2 (double y, double x);

a) The atan2 function returns the arc tangent of x/y  
b) The atan2 function returns the arc tangent of x  
c) The atan2 function returns the arc tangent of y/x  
d) The atan2 function returns the arc tangent of y

2. The cos function computes the cosine of x.  
a) measured in radians  
b) measured in degrees  
c) measured in gradian  
d) measured in milliradian

3. The function computes the hyperbolic cosine of x.  
a) cos(x)  
b) cosine(x)  
c) cosh(x)  
d) cosineh(x)

4. What error occurs if the magnitude of x is too large in sinh(double x)?  
a) domain error  
b) range error  
c) no error  
d) zero is returned

5. Which of the following is the correct code?  
a) tanh(double x)  
b) tanh double x  
c) tanhdouble x  
d) tanhdoublex

6. Name the function that breaks a floating-point number into a normalized fraction and an integral power of 2.  
a) exp()  
b) frexp()  
c) Idexp()  
d) modf()

7. The function computes the exponential function of x.  
a) exp(x)  
b) frexp(x)  
c) frexp x  
d) exp x

8. The ldexp() function multiplies a floating-point number by an integral power of 2.  
a) true  
b) false  
View Answer

9. What will be the output of the following C code?

double log (double -x);

a) returns natural logarithm of x  
b) range error  
c) domain error  
d) returns natural logarithm of -x

10. Which of the given function is a library function under the header math.h?  
a) log10()  
b) log20()  
c) log30()  
d) log50()

**Mathematical Functions – 3:**

1. Which of the following statement is correct?

double x, y, z;

x = 5.123456;

z= modf(x, \*y);

a) y stores integer part of x, z returns fractional part of x  
b) y stores integer part of x, z returns integer part of x  
c) y stores fractional part of x, z returns integer part of x  
d) y stores fractional part of x, z returns fractional part of x

2. A domain error occurs if x is negative and y is not an integral value for the function pow(double x, double y).  
a) true  
b) false

3. A function is declared as sqrt(-x) under the header file math.h, what will the function return?  
a) square root of x  
b) complex number  
c) domain error  
d) range error

4. What will be the output of the following C code?

double x=1.2

printf("%.1lf", ceil(x));

a) 1  
b) 2  
c) 1.0  
d) 2.0

5. What will be the output of the following C code?

double x=3,y= - 6;

printf("%lf %lf", fabs(x), fabs(y));

a) 3 6  
b) -3 6  
c) 3.0 6.0  
d) 3.000000 6.000000

6. What will be the output of the following C code?

double x=1.8;

printf("%.2lf",floor(x));

a) 2.0  
b) 2.00  
c) 1.0  
d) 1.00

7. double \_\_\_\_\_\_ (double x, double y) computes the floating-point remainder of x/y.  
a) modf  
b) fmod  
c) ceil  
d) floor

8. sqrt(x) function is not faster than the apparent equivalent pow(x,0.5).  
a) true  
b) false

9. Which of the given C function can be used instead of the apparent identity pointed to by y?

int x=1;

double y= 0.5 \* (exp (x) + exp (-x));

a) cos(x)  
b) cosh(x)  
c) fmod(x)  
d) modf(x)

10. Which function is used to recombine the fraction and exponent parts of a floating-point value after you have worked on them separately?  
a) frexp()  
b) exp()  
c) modf()  
d) Idexp()

**General Utilities – 1:**

1. \_\_\_\_\_\_\_variable type defined in the header stdlib.h is an integer type of the size of a wide character constant.  
a) size\_t  
b) wchar\_t  
c) div\_t  
d) ldiv\_t

2. Which of the following is the correct description of EXIT\_FAILURE?  
a) This is the value for the exit function to return in case of failure  
b) This is the value for the exit function to terminate the program  
c) This is the value for the exit function to return in case of success  
d) This is the value for the exit function to return in case it is the maximum value

3. RAND\_MAX macro is the maximum value returned by the rand function.  
a) true  
b) false

4. Which of the given function converts the string pointed to, by the argument str to a floating-point number?  
a) atof(const char \*str)  
b) strtod(const char \*str, char \*\*endptr)  
c) atoi(const char \*str)  
d) atol(const char \*str)

5. The\_\_\_\_\_\_\_function converts the initial portion of the string pointed to by, to int representation.  
a) atof()  
b) atoi()  
c) strtod()  
d) atol()  
View Answer

6. atol(const char \*str) Converts the string pointed to, by the argument str.  
a) to a long integer  
b) to a integer  
c) to a floating point number  
d) to a unsigned long integer

7. What will be the output of the following C code?

char str[20];

str= "123546"; res= atof(str);

printf("String value = %s, Float value = %f**\n**", str, res);

a) String value = 123546, Float value = 123546.0  
b) String value = 123546 , Float value = 123546.000000  
c) String value = 123546 , Float value = 0.000000  
d) String value = 123546 , Float value = 123546.000

8. What will be the output of the following C code?

char str[];

strcpy(str, "Hello");

res = atof(str);

printf("String value = %s, Float value = %f**\n**", str, res);

a) String value = Hello, Float value = 0.000000  
b) String value = Hello, Float value = 0  
c) String value = “Hello” , Float value = 0.000000  
d) String value = “Hello” , Float value = 0

9. What will be the output of the following C code?

char str[20];

strcpy(str, "123456");

res = atoi(str);

printf("%s %d**\n**", str, res);

a) 123456 0  
b) 123456 0.0  
c) 123456 123456  
d) 123456 123456.0

10. What will be the output of the following C code?

char str[] ;

strcpy(str, "Hello");

res = atoi(str);

printf(" %s %d**\n**", str, res);

a) Hello 0.000000  
b) “Hello” 0.000000  
c) Hello 0  
d) “Hello” 0

**General Utilities – 2:**

1. What will be the output of the following C code?

char word[20 ] = "1.234555 WELCOME";

char \*w; double dis;

dis= strtod(word, &w);

printf("The number is %lf**\n**", dis);

printf("String is |%s|", w);

a) The number is 1.234555 String is |WELCOME|  
b) The number is 1.2345550 String is |WELCOME|  
c) The number is 1.234555 String is |1.234555 WELCOME|  
d) Errror

2. Which statement is correct work reference to endptr?

double strtod(const char \*nptr, char \*\*endptr);

a) A pointer to the starting string is stored in the object pointed to by endptr, provided that endptr is a null pointer  
b) A pointer to the ﬁnal string is stored in the object pointed to by endptr, provided that endptr is not a null pointer  
c) A pointer to the final string is stored in the object pointed to by endptr, provided that endptr is a null pointer  
d) A pointer to the starting string is stored in the object pointed to by endptr, provided that endptr is not a null pointer

3. Which of the following functions decomposes the input string into three pans: an initial, possibly empty, sequence of white-space characters?  
a) strtod()  
b) atof()  
c) atol()  
d) strtol()

4. The\_\_\_\_\_\_function is used to convert the initial portion of the string pointed to by, to unsigned long int representation.  
a) strtod()  
b) atol()  
c) strtoul()  
d) strtol()

5. Which of the following is the correct syntax of the function strtoul()?  
a) unsigned long int strtoul(const char \*n, char \*\*ptr, int base)  
b) unsigned long int strtoul(const char \*n, char \*\*ptr)  
c) unsigned long int strtoul(const char \*n)  
d) int strtoul(const char \*n)

6. Select the right statement with reference to malloc() and calloc().  
a) malloc() does not set the memory to zero whereas calloc() sets allocated memory to zero  
b) malloc() sets the memory to zero whereas calloc() does not set allocated memory to zero  
c) malloc() sets the memory to zero whereas calloc() sets allocated memory to zero  
d) malloc() does not set the memory to zero whereas calloc() does not set allocated memory to zero

7. The calloc() function allocates space for an array of n objects, each of whose size is defined by size. Space is initialized to all bits zero.  
a) true  
b) false

8. Is this right explanation to the given code?

void \*calloc(size\_t n, size\_t size)

#n -- This is the number of elements to be allocated.

#size -- This is the size of elements.

a) true  
b) false

9. Which among the given function does not return a value?  
a) strtoul()  
b) strtol()  
c) rand()  
d) srand()

10. Which function returns a pseudo-random integer?  
a) srand()  
b) rand()  
c) malloc()  
d) alloc()

**General Utilities – 3:**

1. void free(void \*ptr) deallocates the memory previously allocated by a call to \_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_ or \_\_\_\_\_\_\_\_\_  
a) malloc,getenv,abort  
b) calloc,malloc,exit  
c) calloc,malloc,realloc  
d) exit,getenv,abort

2. The \_\_\_\_\_\_\_\_ function returns no value.  
a) malloc()  
b) realloc()  
c) free()  
d) calloc()

3. What is returned by the function if the space cannot be allocated by the function malloc(), realloc() and calloc()?  
a) value  
b) error  
c) null pointer  
d) no value

4. What is the function of the void \*realloc(void \*str, size-t size);?  
a) allocates space for an array of str objects, each of whose size is size  
b) allocates space for an object whose size is specified by size and whose value is indeterminate  
c) changes the size of the object pointed to by str to the size specified by size  
d) causes the space pointed to by str to be deallocated, that is, made available for further allocation

5. Which among the given function causes abnormal program termination ?  
a) exit()  
b) abort()  
c) atexit()  
d) getenv()

6. Which of the given statement is true with respect to the function atexit()?  
a) The atexit() function cannot return to its caller  
b) The atexit() function is used to return zero if the registration succeeds, nonzero if it fails  
c) The atexit() function returns no value  
d) The atexit() function causes abnormal program termination to occur

7. The behavior is undefined if more than one call to the exit function is executed by a program.  
a) true  
b) false

8. Which function searches an environmenr list that are provided by the host environment?  
a) getenv()  
b) system()  
c) srand()  
d) rand()

9. The system() function passes the string pointed to by string to the host environment to be executed by a command processor in an implementation-defined manner.

int system(const char \*string);

a) true  
b) false

10. What will be the output of the following C code?

int main(void)

{

int rc;

rc = system("time");

exit(0);

}

a) produces error  
b) no value is returned  
c) returns the time  
d) nothing can be said

**General Utilities – 4:**

1. Which of the given function is used for searching?  
a) lsearch()  
b) bsearch()  
c) csearch()  
d) qsearch()

2. Which function is called repeatedly by bsearch() to compare search elements against the elements in the array?  
a) mblem()  
b) wctomb()  
c) compar()  
d) labs()

3. The \_\_\_\_\_\_\_ function sorts an array of objects.  
a) bsort()  
b) hsort()  
c) ssort()  
d) qsort()

4. Choose the correct statement.  
a) bsearch() returns no value to the function  
b) getenv() returns no value to the function  
c) qsort() returns no value to the function  
d) realloc() returns no value to the function

5. Which statement is true regarding abs() and labs()?  
a) The abs() function is similar to the labs() function, except that the argument and the returned value each of them have type long int  
b) The abs() function is not similar to the labs() function, except that the argument and the returned value in both functions have type long int  
c) The abs() function is similar to the labs() function, except that the argument and the returned value each have type short int  
d) The abs() function is not similar to the labs() function, except that the argument and the returned value in both function have type short int

6. The abs() function computes the absolute value \_\_\_\_\_\_\_\_\_\_  
a) a floating number  
b) an integer number  
c) a double number  
d) all of the mentioned

7. Which function will return the quotient and remainder on division of numerator with denominator?  
a) div()  
b) div\_t()  
c) ldiv\_t()  
d) labs()

8. What members do the structure returned by function div() contains?  
a) int quot and int rem  
b) float quot and float rem  
c) double quot and double rem  
d) no members are returned by div()

9. Which of the given structure is returned by the function ldiv()?  
a) div\_t  
b) ldiv\_t  
c) div\_i  
d) ldiv\_i

10. Select the multibyte character function defined under the header file stdlib.h.  
a) wctomb()  
b) mblen()  
c) mbtowc()  
d) all of the mentioned

**General Utilities – 5:**

1. The number of bytes contained in the multibyte character pointed to by a character is \_\_\_\_\_\_\_\_\_\_\_  
a) getenv()  
b) bsearch()  
c) mblen()  
d) qsort()

2. The pointer used in the mblen() function points to the \_\_\_\_\_\_\_\_\_  
a) first byte of multibyte character  
b) last byte of multibyte character  
c) middle byte of multibyte character  
d) no pointer is used in mblen function

3. What is the name of the function that is used to convert multibyte character to wide character?  
a) mblen()  
b) mbtowc()  
c) mbstowcs()  
d) wcstombs()

4. Which function converts the wide-character string to a multibyte string?  
a) wcstombs()  
b) mbstowcs()  
c) mbtowc()  
d) mblen()

5. The C library function \_\_\_\_\_\_\_\_\_\_\_\_\_function converts the wide character to its multibyte representation.  
a) mblen()  
b) mbtowc()  
c) wcstombs()  
d) wctomb()

6. The mbstowcs() function is used to return the number of array elements modified, not including a terminating zero code, if any.  
a) true  
b) false

7. What will the given C code do?

#include <stdlib.h>

\_Mbsave\_Mbxlen={0};

int (mblen)(const char \*s ,size\_t n)

{

return(\_Mbtowc(NULL s,n,&\_Mbxlen));

}

a) determine length of next multibyte code  
b) determine next multibyte code  
c) translate multibyte string to wide char string  
d) translate wide character to multibyte string

8. What is the purpose of the given C code?

#include <stdlib.h>

\_Mbsave \_Mbxtowc = {0};

int (mbtowc) (wchar\_t \*pwc, const char \*a, size\_t n)

{

return (-Mbtowc (pwc, s, n, &-Mbxtowc) ) ;

}

a) determine length of next multibyte code  
b) translates multibyte character to wide character  
c) translate multibyte string to wide char string  
d) translate wide character to multibyte string

9. What is “a” in the given C code?

size\_t wcstombs(char \*s, const wchar\_t \*a, size\_t n)

a) “a” is wide character string to be converted  
b) “a” is pointer to an array of char elements  
c) “a” is pointer to the first byte of a multi-byte character  
d) “a” C multibyte character string to be interpreted

10. mblen() function returns 0,if a null wide character was recognized. It returns -1 if an invalid multi-byte sequence was encountered.  
a) true  
b) false

**General Utilities – 6:**

1. What will the code display on compiling?

void funccall()

{

printf("this is funccall**\n**");

}

void main ()

{

atexit(funccall);

printf("program starts**\n**");

printf("program ends**\n**");

}

a)

program starts

this is funccall

program ends

b)

this is funccall

program starts

program ends

c)

program starts

program ends

this is funccall

d) error

2. What will be the output of the following C code?

int main ()

{

printf("starting of program**\n**");

printf("program exits**\n**");

exit(0);

printf("program ends**\n**");

return(0);

}

a)

starting of program

program exits

program ends

b)

starting of program

program exits

c)

starting of program

program ends

d) error

3. What will the following C code do on compilation?

void main ()

{

char com[50];

strcpy( com, "dir" );

system(com);

}

a) error  
b) lists down all the files and directories in the current directory under windows machine  
c) terminates the calling process immediately  
d) calls specified function and terminates it at the end of the program

4. What will be the output of the following C code?

void main()

{

div\_t res;

res = div(34, 4);

printf("quotient part = %d**\n**", res.quot);

printf("remainder part = %d**\n**", res.rem);

}

a)

quotient part=0

remainder part=4

b)

quotient part=8

remainder part=2

c)

quotient part=4

remainder part=0

d)

quotient part=2

remainder part=8

5. Initial seed is \_\_\_\_\_\_\_\_ for the function srand(unsigned int seed).  
a) 0  
b) 1  
c) 00  
d) 01

6. Which statement is true with respect to RAND\_MAX?  
a) specifies value for status argument to exit indicating failure  
b) specifies value for status argument to exit indicating success  
c) specifies maximum value returned by rand()  
d) specifies maximum value returned by srand()

7. Which of the given function differs from the statement’errno is not neccessarily set on conversion error’?  
a) atof()  
b) atoi()  
c) atol()  
d) strtod()

8. void free(void \*p) performs which of the following functions?  
a) returns pointer to allocated space for existing contents of p  
b) de-allocates space to which p points  
c) to abnormally terminate the program  
d) no such function defined in stdlib.h

9. Which of the given option is declared under the header file stdlib.h?  
a) SEEK\_CUR  
b) SEEK\_END  
c) CLOCKS\_PER\_SEC  
d) EXIT\_SUCCESS

10. MB\_CUR\_MAX is not defined in stdlib.h.  
a) true  
b) false

**Diagnostics – 1:**

1. The header file assert.h of the C Standard Library defines \_\_\_\_\_\_\_\_ macro.  
a) stderr  
b) stdarg  
c) setjmp  
d) assert

2. What is the name of the macro that is referred by assert macro defined in assert .h?  
a) STDARG  
b) SETJMP  
c) NDEBUG  
d) STDERR

3. If NDEBUG is defined as a macro name, at the point where <assert.h> is included, then assert macro is defined as #define assert(ignore) ((void)0).  
a) true  
b) false

4. The assert shall be implemented as a \_\_\_\_\_\_ not as an actual \_\_\_\_\_\_\_\_  
a) function, macro  
b) macro, function  
c) header, macro  
d) macro, header

5. The assert macro returns\_\_\_\_\_\_\_\_\_\_value.  
a) integer  
b) float  
c) double  
d) no

6. The macro void assert(int expression) allows the diagnostic information to be written in which of the following files?  
a) standard error file  
b) output file  
c) string file  
d) character file

7. Which is the correct declaration of macro assert?  
a) void assert(int expression);  
b) void assert(float expression);  
c) void assert(double expression);  
d) void assert( expression);

8. If the expression in void assert(int expression) is zero then a message is printed on stderr(standard error file).  
a) true  
b) false

9. void assert(int expression) when the expression is evaluated to true?  
a) assert returns integer value  
b) assert displays error message  
c) assert returns nothing  
d) assert displays pattern

10. Which function is called by macro assert to terminate the execution?  
a) exit()  
b) atexit()  
c) abort()  
d) no function called

**Diagnostics – 2:**

1. The following message is displayed in stderr.  
Assertion failed: expression, file filenum, line nmn  
a) true  
b) false

2. The source filename and line number come from the preprocessor macros \_\_\_\_\_\_\_\_ and \_\_\_\_\_\_  
a) \_ \_FILE\_ \_ and \_ \_LINE\_ \_  
b) \_ \_NAME\_ \_ and \_ \_NUMBER\_ \_  
c) \_ \_FILENAME\_ \_ and \_ \_NMN\_ \_  
d) \_ \_FILE\_ \_ and \_ \_NUM\_ \_

3. The function abort() is defined in which of the following header file?  
a) stdio.h  
b) stdarg.h  
c) stdlib.h  
d) assert.h

4. Correct code to turn assertions ON at various places throughout a source file is \_\_\_\_\_

#undef NDEBUG

#include <assert.h>

b)

#define NDEBUG

#include <assert.h>

c)

#define NDEBUG

#include <stdlib.h>

d)

#undef NDEBUG

#include <stdlib.h>

5. Correct code to turn assertions OFF at various places throughout a source file is \_\_\_\_\_\_\_\_\_  
a)

#undef NDEBUG

#include <assert.h>

b)

#define NDEBUG

#include <assert.h>

c)

#define NDEBUG

#include <stdlib.h>

d)

#undef NDEBUG

#include <stdlib.h>

6. Which line from the given code is the passive form?

#undef assert

#ifdef NDEBUG

#define assert (test) ( (void) 0)

#else

#define assert (test)

#endif

a) #define assert(test)  
b) #define assert(test) ((void)0)  
c) #ifdef NDEBUG  
d) #undef assert

7. What will be the output of the following C code?

#include <assert.h>

#include <stdio.h>

#include <stdlib.h>

void Assert (char \*mesg)

{

fputs (mesg, stderr);

fputs (" -- assertion failed**\n**" , stderr);

abort () ;

}

a) prints only assertion message  
b) program is just aborted  
c) prints assertion message and aborts  
d) no action takes place

8. Which macro can be used to detect and report exceptional conditions?  
a) extern  
b) edom  
c) assert  
d) lbdl\_min 1e-37

9. What will be the output of the following C code?

#include <assert.h>

#include <stdio.h>

void main()

{

int n=11;

char str[50]="program";

assert(n >= 10);

printf(" output: %d**\n**", n);

assert(str != NULL);

printf("output: %s**\n**", str);

}

a) output: 11  
b) error message  
c)

output: 11

output: program

d) output: program

10. What will be the output of the following C code?

#include <assert.h>

#include <stdio.h>

void main()

{

int n=12;

char str[50]="";

assert(n >= 10);

printf(" output: %d**\n**", n);

assert(str != NULL);

printf("output: %s**\n**", str);

}

a)

output: 12

output:

b) output: 12  
c)

output: 12

assertion error

d) error

**Variable Argument Lists:**

1. How many macros are defined in the header file stdarg.h?  
a) one  
b) two  
c) three  
d) four

2. The header file stdarg.h defines a variable type \_\_\_\_\_\_\_\_  
a) va\_list  
b) v\_list  
c) size\_t  
d) var\_list

3. The three macros defined by stdarg.h is \_\_\_\_\_\_\_\_\_  
a) start(), arg() and end()  
b) var\_start(), var\_arg() and var\_end()  
c) va\_start(), va\_arg() and va\_end()  
d) v\_start(), v\_arg() and v\_end()

4. If access to the varying arguments is desired then the called function shall declare \_\_\_\_\_\_\_\_ having type va\_list.  
a) class  
b) object  
c) function  
d) variable

5. Which macro retrieves the next argument in the parameter list of the function with type type?  
a) type va\_arg(va\_list ap, type)  
b) type var\_arg(va\_list ap, type)  
c) type v\_arg(va\_list ap, type)  
d) type val\_arg(va\_list ap, type)

6. The \_\_\_\_\_\_\_ macro shall be invoked before any access to the unnamed arguments.  
a) va\_arg  
b) va\_end  
c) va\_list  
d) va\_start

7. \_\_\_\_\_\_ macro must be called before using \_\_\_\_\_\_ and \_\_\_\_\_\_\_\_  
a) va\_arg, va\_end and va\_start  
b) va\_start, va\_end and va\_arg  
c) va\_end, va\_arg and va\_start  
d) v\_arg, v\_end and v\_start

8. The C library macro type \_\_\_\_\_\_\_\_\_ retrieves the next argument in the parameter list of the function with type.  
a) va\_end  
b) va\_arg  
c) va\_start  
d) no macros

9. What is the role of the given C function?

void va\_end(va\_list ap)

a) allows a function with variable arguments which used the va\_start macro to return  
b) retrieves the next argument in the parameter list  
c) initializes ap variable to be used with the va\_arg and va\_start macros  
d) returns the next additional argument as an expression

10. Which header file should be included if a function has to be defined such that it can accept variable number of arguments?  
a) stdlib.h  
b) stdarg.h  
c) assert.h  
d) setjmp.h

**Localization:**

1. Which header file is used to define data formats and currency symbols?  
a) setjmp.h  
b) locale.h  
c) stdarg.h  
d) assert.h

2. Which among the given macros is defined in the header file locale.h?  
a) SCHAR\_MAX  
b) FLT\_RADIX 2  
c) EDOM  
d) LC\_CTYPE

3. Which macro sets everything defined under locale. h?  
a) LC\_ALL  
b) LC\_COLLATE  
c) LC\_SET  
d) LC\_TIME

4. Select the function that reads or sets location dependent information.  
a) longjmp()  
b) setlocale()  
c) assert()  
d) toupper()

5. Select the correct statement.  
a) LC\_MONETARY affects the monetary information  
b) LC\_MONETARY does not affect the monetary information  
c) LC\_ALL does not set everything  
d) LC\_CTYPE affects only one character functions

6. Which macro is used in the setlocale() function?  
a) LC\_SET  
b) FLT\_RADIX 2  
c) LC\_MESSAGES  
d) SHRT\_MAX

7. LC\_COLLATE affects strcoll() and strxfrm() functions.  
a) true  
b) false

8. Which macro affects the strftime() function?  
a) LC\_TIME  
b) LC\_SEC  
c) LC\_MIN  
d) LC\_SET

9. Select the macro that affects the information provided by localeconv function.  
a) LC\_ALL  
b) LC\_COLLATE  
c) LC\_NUMERIC  
d) LC\_CTYPE

10. What is returned by the function localeconv()?  
a) current location value  
b) past location value  
c) pointer to the last location  
d) pointer to the current location

**Non-Local Jumps – 1:**

1. Which of the following header file defines one function longjmp(), and one variable type jmp\_buf?  
a) stdarg.h  
b) locale.h  
c) setjmp.h  
d) stdlib.h

2. Which of the given options is an array type used for holding information?  
a) longjmp  
b) setjmp  
c) jmp\_buf  
d) no such variable

3. Which macro saves the current environment into the variable environment for later use by the function longjmp().  
a) setjmp  
b) longjmp  
c) jmp  
d) set\_jmp

4. If setjmp() macro returns directly from the macro invocation, it\_\_\_\_\_\_  
a) returns zero  
b) returns non-zero  
c) produces error  
d) nothing can be said

5. A non-zero value is returned, if setjmp() returns from a longjmp() function call.  
a) false  
b) true

6. Select the correct declaration of setjmp().  
a) int setjmp(jmp\_buf environment)  
b) int setjmp(long\_jmp environment)  
c) int setjmp(jmp\_buf )  
d) int setjmp(long\_jmp)

7. How many times can the macro setjmp() return?  
a) one time  
b) two times  
c) three times  
d) many times

8. longjmp() function is the only function defined under the header file setjmp.h?  
a) true  
b) false

9. Which function restores the environment saved by the most recent invocation of the setjmp() macro in the same invocation of the program?  
a) jmp\_buf  
b) longjmp  
c) jmpbuf  
d) long\_jmp

10. Choose the right declaration of longjmp() function.  
a) void longjmp(jmp\_buf environment, int value)  
b) void longjmp(setjmp environment, int value)  
c) void longjmp(int value, jmp\_buf environment)  
d) void longjmp(int value, setjmp environment)

**Non-Local Jumps – 2:**

1. The header file setjmp.h is used to \_\_\_\_\_\_\_\_\_\_  
a) set location specific information  
b) control low-level calls and returns to and from functions  
c) handle signals reported during a program’s execution  
d) manipulate strings (character arrays)

2. The void longjmp( jmp-buf env, int val) function causes setjmp() macro to return \_\_\_\_\_\_\_ value; if val is 0.  
a) zero  
b) one  
c) null  
d) no return

3. Which is the true statement with respect to the function longjmp()?  
a) the function where setjmp() was called has terminated, then the results are undefined  
b) the function where setjmp() was called has terminated, then the results are defined  
c) the function where jmp\_buf was called has terminated, then the results are undefined  
d) the function where jmp\_buf was called has terminated, then the results are defined

4. Which of the given statement is not true with respect to void longjmp( jmp-buf env, int val)?  
a) The variable value cannot be zero  
b) env is the object containing information to restore the environment at the jmp\_buf’s calling point  
c) This function does not return any value  
d) This function restores the environment saved by the most recent call to setjmp() macro

5. What is the function of the given longjump(jmp\_buf buf, i)?  
a) go back to place buf is pointing to and return i  
b) go back to place buf is pointing to and return 0  
c) uses buf to remember current position and returns 0  
d) uses buf to remember current position and returns i

6. How many times does the function longjmp() returns?  
a) once  
b) twice  
c) thrice  
d) never

7. A less common use of setjmp.h is to create syntax similar to \_\_\_\_\_\_\_\_\_\_\_\_  
a) errno  
b) variable arguments  
c) coroutines  
d) retval

8. What will the following C statement do?

#include < setjmp.h >

int setjmp(jmp\_buf env);

a) save the current state of the registers into env  
b) resets the registers to the values saved in env  
c) provides a definition for env structure  
d) accept variable(env) argument lists to be written

9. What are the contents of the register?  
a) sp, fp only  
b) sp only  
c) fp, pc only  
d) sp, fp, pc

10. setjmp takes a jmp\_buf type and different other type variables as input.  
a) true  
b) false

**Signal Handling:**

1. Select the right statement.  
a) synchronous signal occurs because of the action that your program takes  
b) synchronous signal occurs because of action outside your program  
c) asynchronous signal occurs because of the action that your program takes  
d) division by zero is asynchronous

2. What does raise functions declared in signal.h do?  
a) reports a synchronous signal  
b) let’s you specify handling of signals  
c) reports a asynchronous signal  
d) doesn’t let you specify handling of signals

3. What is the type declared by the header file signal.h?  
a) sig\_atomic\_t  
b) sig\_signal\_t  
c) sig\_signal\_h  
d) sig\_stomic\_h

4. Which among the given header file is used to handle different signals reported during program execution?  
a) stdarg.h  
b) assert.h  
c) signal.h  
d) setjmp.h

5. Select the macro that abnormally terminates the program.  
a) SIGILL  
b) SIGTERM  
c) SIGABRT  
d) SIGFPE

6. Which of the following is the correct description of the macro SIGFPE?  
a) erroneous arithmetic operation such as zero divide  
b) invalid access to storage  
c) termination request sent to the program  
d) receipt of the interactive attention signal

7. \_\_\_\_\_\_\_\_\_ gives receipt of an interactive attention signals.  
a) SIGILL  
b) SIGTERM  
c) SIGINT  
d) SIGFPE

8. The sig argument specifies the signal, which may be any signal except \_\_\_\_\_\_\_ and \_\_\_\_\_\_\_\_\_  
a) SIG\_DFL, SIG\_IGN  
b) SIGKILL, SIGSTOP  
c) SIG\_KILL, SIG\_STOP  
d) SIGCHLD, SIG\_IGN

9. void (\*signal(int sig, void (\*func)(int)))(int);If the value of func is SIG\_IGN then \_\_\_\_\_\_\_\_\_  
a) the signal will be ignored  
b) default handling for that signal will occur  
c) The signal() function will fail to execute  
d) the signal will be ignored

10. In the c library function void (\*signal(int sig, void (\*func)(int)))(int), which statement is true with respect to func?  
a) func is a pointer to the function  
b) func is pointer to sig  
c) func is a static variable  
d) func is a pointer that points to all type of data

**Standard Definition:**

1. Some types and macros defined under the header file stddef.h may be defined under other header files too.  
a) True  
b) False

2. size\_t is of \_\_\_\_\_\_\_\_\_\_\_\_\_\_ type.  
a) signed integer  
b) signed character  
c) unsigned integer  
d) unsigned character

3. Which of the following returns a signed integer type on finding the difference between two pointers to elements in the same array?  
a) \_\_cptrdiff\_\_  
b) cptrdiff\_t  
c) \_\_ptrdiff\_\_  
d) ptrdiff\_t

4. What will be the output of the following C code?

#include <stddef.h>

int main(void)

{

int num[10];

int \*p1=&num[14], \*p2=&num[19];

ptrdiff\_t a = p1-p2;

printf("%d", a);

}

a) 5  
b) -5  
c) error  
d) 10

5. What will be the output of the following C code?

#include <stddef.h>

int main(void)

{

int num[15];

int \*p1=&num['a'], \*p2=&num['A'];

ptrdiff\_t d = p1-p2;

printf("%d", d);

}

a) 15  
b) -32  
c) 32  
d) error

6. Which of the following is not defined under the header file stddef.h?  
a) size\_t  
b) ptrdiff\_t  
c) exp\_t  
d) null

7. Point out the error (if any) in the following C code?

#include <stdlib.h>

#include <stdio.h>

int main(void)

{

int\* p = NULL;

struct S \*s = NULL;

void(\*f)(int, double) = NULL;

char \*ptr = malloc(15);

if (ptr == NULL) printf("Out of memory");

free(ptr);

}

a) Error in the statement: void(\*f)(int, double) = NULL;  
b) Error in the statement: char \*ptr = malloc(15);  
c) Error in the statement: struct S\*s = NULL;  
d) No error

8. A type whose alignment requirement is at least as large as that of every data type \_\_\_\_\_\_\_\_\_\_\_\_  
a) max\_align\_t  
b) ptrdiff\_t  
c) size\_t  
d) null

9. The macro offset expands to a constant of type \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
a) size\_t  
b) print\_t  
c) ptrdiff\_t  
d) null

10. When we use multiple alignas specifiers in the same declaration, the \_\_\_\_\_\_\_\_\_\_\_\_ one is used.  
a) first  
b) strictest  
c) last  
d) middle

**Date and Time Functions – 1:**

1. Which of the following library functions returns the time in UTC (Greenwich mean time) format?  
a) localtime()  
b) gettime()  
c) gmtime()  
d) settime()

2. What will be the output of the following C code? (By date we mean: date, month and year)

advertisement

#include<stdio.h>

#include<stdlib.h>

#include<time.h>

int main()

{

time\_t ct;

time(&ct);

printf("%s**\n**",ctime(&ct));

}

a) only current date  
b) only current date and current time  
c) current date, current time and the day of the week  
d) only current time

3. What will be the output of the following C code if the current system date is 6/22/2017?

#include<stdio.h>

#include<stdlib.h>

#include<time.h>

int main()

{

time\_t ct;

time(&ct);

struct tm \*mt=localtime(&ct);

printf("%d**\n**",mt-> tm\_mon+2);

}

a) 8  
b) 7  
c) 5  
d) 6

4. What will be the output of the following C code if the current system date is 6/22/2017?

#include<stdio.h>

#include<stdlib.h>

#include<time.h>

typedef struct tm tm;

int main()

{

time\_t ct;

time(&ct);

tm \*mt=localtime(&ct);

printf("%d**\n**",mt-> tm\_year);

}

a) 17  
b) 2017  
c) error  
d) 117

5. What will be the output of the following C code if the current system date is 6/22/2017?

#include<stdio.h>

#include<stdlib.h>

#include<time.h>

int main()

{

time\_t ct;

time(&ct);

struct tm \*mt=localtime(&ct);

printf("%d**\n**",mt-> tm\_date);

}

a) 22  
b) 6  
c) 22/6  
d) error

6. The purpose of the function ctime() is that \_\_\_\_\_\_\_\_\_\_\_  
a) it returns a string representing the local time  
b) it returns a void string  
c) it returns a string representing the time in UTC format  
d) it returns a string representing the time stored in a structure

7. What will be the output of the following C code?

#include<time.h>

int main (void)

{

float n = time(NULL);

printf("%.2f**\n**" , n);

}

a) time in seconds from 1 January, 1970  
b) time in minutes from 1 January, 1970  
c) time in seconds from 1 January, 1980  
d) time in minutes from 1 January, 1980

8. What will be the output of the following C code if the system date is 6/2/2017(Friday)?

#include<stdio.h>

#include<time.h>

int main()

{

struct tm \*local, \*gm;

time\_t t;

t=time(NULL);

local=localtime(&t);

printf("%d",local->tm\_wday);

return 0;

}

a) 6  
b) 5  
c) error  
d) 0

9. Which of the following functions returns a pointer to a string representing the date and time stored in a structure?  
a) ctime()  
b) time()  
c) asctime()  
d) localtime()

10. What will be the output of the following C code if it is executed on 2nd January, 2017 (system date)?

#include<stdio.h>

#include<time.h>

int main()

{

struct tm \*local, \*gm;

time\_t t;

t=time(NULL);

local=localtime(&t);

printf("%d",local->tm\_yday);

return 0;

}

1. 0

b) 1  
c) 2  
d) Error

**Date and Time Function – 2:**

1. Which of the following format specifiers is used to represent the name of the time zone?  
a) %A  
b) %B  
c) %H  
d) %Z

2. What will be the output of the following C code if the system time is 4:27 PM?

#include<stdio.h>

#include<time.h>

int main()

{

struct tm \*ptr;

time\_t t;

char str[100];

t = time(NULL);

ptr = localtime(&t);

strftime(str,100,"%H %p %M minutes",ptr);

puts(str);

return 0;

}

a) 16 27 minutes  
b) 4 27 minutes  
c) 16 PM 27 minutes  
d) 4 PM 27 minutes

3. What will be the output of the following C code if the system date is 8/22/2016?

#include<stdio.h>

#include<time.h>

int main()

{

struct tm \*ptr;

time\_t t;

char str[100];

t = time(NULL);

ptr = localtime(&t);

strftime(str,100,"%B",ptr);

puts(str);

return 0;

}

a) 9  
b) August  
c) Aug  
d) Error

4. What will be the output of the following C code if the system date is 6/2/2017 (Friday)?

#include<stdio.h>

#include<time.h>

int main()

{

struct tm \*ptr;

time\_t t;

char str[100];

t = time(NULL);

ptr = localtime(&t);

strftime(str,100,"%A",ptr);

puts(str);

return 0;

}

a) Error  
b) Fri  
c) Friday  
d) 6

5. Which of the following library functions is used to read location dependent information?  
a) localtime()  
b) localeconv()  
c) localcon()  
d) local()

6. Which of the following functions is used to convert the date and time into a calendar format?  
a) difftime()  
b) clock()  
c) mktime()  
d) ctime()

7. What will be the output of the following C code?

#include<stdio.h>

#include<time.h>

main()

{

struct tm t;

time\_t tc;

t.tm\_year=2017-1900;

t.tm\_mday=25;

t.tm\_mon=5;

t.tm\_hour=1;

t.tm\_min=30;

t.tm\_sec=0;

t.tm\_isdst=0;

tc=mktime(&t);

printf(ctime(&tc));

}

a) Sun Jun 25 01:30:00 2017  
b) Sun June 25 1:30 2017  
c) Sun Jun 25 1:30 117  
d) Sun June 25 1:30:00 117

8. The value of tm\_isdst is \_\_\_\_ when DST( Daylight Savings Time) is in effect, \_\_\_\_\_\_ when DST is not in effect and \_\_\_\_\_\_ when the DST status is unknown.  
a) -1, 1, 0  
b) 1, 0, -1  
c) 0, 1, -1  
d) 1, -1, 0

9. The library function clock() returns the number of \_\_\_\_\_\_\_\_\_ elapsed since the start of the program.  
a) minutes  
b) clock ticks  
c) milli-seconds  
d) micro-seconds

10. What will be the output of the following C code if the name entered is “TOM” and time taken to enter this name is 2 seconds?

#include <stdio.h>

#include <time.h>

int main ()

{

time\_t time1,time2;

char get\_input [256];

double dif\_sec;

time (&time1);

printf ("Please enter the name of your pet: ");

gets (get\_input);

time (&time2);

dif\_sec = difftime (time2,time1);

printf ("%.2f**\n**", dif\_sec );

return 0;

}

a) Error  
b) 2  
c) 2.0  
d) 2.00

**Date and Time Function – 3:**

1. What will be the output of the following C code, if the system date is 6/23/2017?

#include<stdio.h>

#include<time.h>

int main()

{

struct tm \*local;

time\_t t;

t=time(NULL);

local=localtime(&t);

printf("%d",local->tm\_mday);

return 0;

}

a) 6  
b) 22  
c) 23  
d) error

2. What will be the output of the following C code, if the system date is 6/24/2017?

#include<stdio.h>

#include<stdlib.h>

#include<time.h>

int main()

{

time\_t ct;

time(&ct);

printf("%s**\n**",(&ct));

}

a) Error  
b) Junk value  
c) 6  
d) June

3. What is the meaning of the following C code if output is 0?

#include<stdio.h>

#include<time.h>

int main()

{

struct tm \*local;

time\_t t;

t=time(NULL);

local=localtime(&t);

printf("%d",local->tm\_isdst);

return 0;

}

a) DST is in effect  
b) DST status is unknown  
c) DST is not in effect  
d) DST is corresponding with local time

4. The following C code results in an error. State whether true or false.

#include<stdio.h>

#include<time.h>

int main()

{

struct tm \*local;

time\_t t;

t=time(NULL);

local=asctime(localtime(&t));

printf("%d",local->tm\_wday);

return 0;

}

a) True  
b) False

5. Which of the following format specifiers is used to specify whether the given time is in AM or PM?  
a) %P  
b) %I  
c) %X  
d) %p

6. What will be the output of the following C code if the system time is 1:52 PM (Sunday)?

#include<stdio.h>

#include<time.h>

int main()

{

struct tm \*ptr;

time\_t t;

char str[100];

t = time(NULL);

ptr = localtime(&t);

strftime(str,100,"%I",ptr);

puts(str);

return 0;

}

a) 13  
b) 01  
c) 1  
d) error

7. Which of the following format specifiers is used to represent the hours in the 24 hour clock (0-23) format?  
a) %I  
b) %H  
c) %i  
d) %h

8. What will be the output of the following C code?

#include <stdio.h>

#include <time.h>

int main ()

{

double d;

d = difftime (5,17);

printf ("%.2f**\n**", d );

return 0;

}

a) 12.00  
b) -12.00  
c) Error  
d) 12

9. The value returned by the library function mktime(), on failure is \_\_\_\_\_\_\_\_\_  
a) -1  
b) 0  
c) 1  
d) -2

10. Which of the following is defined under the header file time.h?  
a) strnct()  
b) fabs()  
c) iscntrl()  
d) null

**Implementation-Defined Limits – 1:**

1. The maximum value of a signed char is not the same as the maximum value of an unsigned char.  
a) True  
b) False

2. What will be the output of the following C code?

advertisement

#include <stdio.h>

#include <limits.h>

int main()

{

printf("The minimum value of LONG = %lf**\n**", LONG\_MIN);

return 0;

}

a) -2147483648 (The minimum value of LONG)  
b) 0.000000  
c) 0  
d) error

3. The macro \_\_\_\_\_\_\_\_\_\_ defines the number of bits in a byte, which is equal to \_\_\_\_\_\_\_  
a) CHAR\_BIT, 4  
b) CHAR\_BYTE, 8  
c) CHAR\_BIT, 8  
d) CHAR\_BYTE, 4

4. What will be the output of the following C code?

#include<stdio.h>

#include<limits.h>

main()

{

printf("%f",FLT\_MIN);

}

a) Minimum negative value of float  
b) Maximum positive value of float  
c) Error  
d) Minimum positive value of float

5. The macro MB\_LEN\_MAX is used to find \_\_\_\_\_\_\_\_\_  
a) Maximum number of bytes in a multi-byte character  
b) Whether the given function is valid or not  
c) The maximum time taken for the execution of a particular function  
d) Maximum number of bits in a multi-byte character

6. The value of CHAR\_MAX will be equal to SCHAR\_MAX when \_\_\_\_\_\_\_\_\_  
a) char represents positive value  
b) char represents value equal to 0  
c) char represents negative value  
d) char represents an exponential value

7. What will be the output of the following C code, given that the maximum value of signed char is 127 and that of unsigned char is 255.

#include<stdio.h>

#include<limits.h>

#define SCHAR\_MAX

main()

{

printf("%d",SCHAR\_MAX+1);

}

a) 256  
b) Error  
c) 1  
d) 128

8. The macro which is used to find the maximum value of an unsigned integer is \_\_\_\_\_\_\_\_\_  
a) UNINT\_MAX  
b) UNSINT\_MAX  
c) UINT\_MAX  
d) UNIT\_MAX

9. To find the maximum value of an object of type unsigned long long int, we use the macro \_\_\_\_\_\_\_\_\_  
a) ULINT\_MAX  
b) ULLINT\_MAX  
c) ULONG\_MAX  
d) ULLONG\_MAX

10. Which of the following macros is defined under the header limits.h?  
a) FLT\_ROUNDS  
b) USHRT\_MAX  
c) DBL\_MAX  
d) DECIMAL\_DIG

**Implementation-Defined Limits – 2:**

1. What will be the output of the following C code?

#include<stdio.h>

#include<limits.h>

main()

{

if(UCHAR\_MAX<=SCHAR\_MAX)

printf("hello");

else

printf("good");

}

a) error  
b) hello  
c) good  
d) hellogood

2. Which of the following macros is not defined?  
a) ULONG\_MIN  
b) LONG\_MIN  
c) ULONG\_MAX  
d) LONG\_MAX

3. Given that the value of SHRT\_MAX is equal to 32767 and that of SHRT\_MIN is equal to -32768, What will be the output of the following C code?

#include<stdio.h>

#include<limits.h>

main()

{

int d;

d=SHRT\_MAX + SHRT\_MIN+1;

printf("%d",d);

}

a) -1  
b) error  
c) 1  
d) 0

4. What will be the output of the following C code?

#include<stdio.h>

#include<limits.h>

main()

{

int d;

d=CHAR\_MIN;

printf("%d",d);

}

a) 0  
b) -128  
c) error  
d) -1

5. \_\_\_\_\_\_\_\_\_\_\_\_\_ defines the minimum value for a short integer.  
a) SHINT\_MIN  
b) SHRT\_MIN  
c) SINT\_MIN  
d) SHORT\_MIN

6. The macro definition of INT\_MIN is \_\_\_\_\_\_\_\_\_\_\_\_  
a) –INT\_MAX – 1  
b) INT\_MAX – 1  
c) –INT\_MAX + 1  
d) INT\_MAX + 1

7. What will be the output of the following C code?

Maximum value of int = 1000

Maximum value of float = 5000

Maximum value of short int = 327

Minimum value of short int = -328

#include<stdio.h>

#include<limits.h>

#include<float.h>

main()

{

short int d;

d=INT\_MAX + FLT\_MAX;

printf("%d",d);

}

a) error  
b) 6000  
c) 327  
d) -328

8. The macros defined under the header file limits.h are not defined under any other header file.  
a) False  
b) True

9. What will be the output of the following C code if the value of UCHAR\_MAX is 127?

#include<stdio.h>

#include<limits.h>

int main()

{

int d;

d=CHAR\_MAX;

printf("%c",d);

}

a) Error  
b) 127  
c) Alphabet corresponding to the value 127  
d) Junk value

10. The minimum value of CHAR\_BIT is equal to \_\_\_\_\_\_\_\_  
a) 2  
b) 4  
c) 8  
d) 16

**C MCQs on Dynamic Memory Allocation**

**Static vs Dynamic Memory Allocation:**

1. Local variables are stored in an area called \_\_\_\_\_\_\_\_\_\_\_  
a) Heap  
b) Permanent storage area  
c) Free memory  
d) Static

2. The size of both stack and heap remains the same during run time.  
a) True  
b) False

3. Choose the statement which is incorrect with respect to dynamic memory allocation.  
a) Memory is allocated in a less structured area of memory, known as heap  
b) Used for unpredictable memory requirements  
c) Execution of the program is faster than that of static memory allocation  
d) Allocated memory can be changed during the run time of the program based on the requirement of the program

4. Which of the following header files must necessarily be included to use dynamic memory allocation functions?  
a) stdlib.h  
b) stdio.h  
c) memory.h  
d) dos.h

5. The type of linked list in which the node does not contain any pointer or reference to the previous node is \_\_\_\_\_\_\_\_\_\_\_\_\_  
a) Circularly singly linked list  
b) Singly linked list  
c) Circular doubly linked list  
d) Doubly linked list

6. Which of the following is an example for non linear data type?  
a) Tree  
b) Array  
c) Linked list  
d) Queue

7. Queue data structure works on the principle of \_\_\_\_\_\_\_\_\_\_\_\_  
a) Last In First Out (LIF0)  
b) First In Last Out (FILO)  
c) First In First Out (FIFO)  
d) Last In Last Out (LILO)

8. Which of the following is an example of static memory allocation?  
a) Linked list  
b) Stack  
c) Queue  
d) Array

9. Array is preferred over linked list for the implementation of \_\_\_\_\_\_\_\_  
a) Radix sort  
b) Insertion sort  
c) Binary search  
d) Polynomial evaluation

10. The advantage of using linked lists over arrays is that \_\_\_\_\_\_\_\_  
a) Linked list is an example of linear data structure  
b) Insertion and deletion of an element can be done at any position in a linked list  
c) Linked list can be used to store a collection of homogenous and heterogeneous data types  
d) The size of a linked list is fixed

**DMA Functions, Memory Leak, Dangling Pointers – 1:**

1. What will be the output of the following C code if the input entered as first and second number is 5 and 6 respectively?

#include<stdio.h>

#include<stdlib.h>

main()

{

int \*p;

p=(int\*)calloc(3\*sizeof(int));

printf("Enter first number**\n**");

scanf("%d",p);

printf("Enter second number**\n**");

scanf("%d",p+2);

printf("%d%d",\*p,\*(p+2));

free(p);

}

a) 56  
b) Address of the locations where the two numbers are stored  
c) 57  
d) Error

2. In the function malloc(), each byte of allocated space is initialized to zero.  
a) True  
b) False

3. Which of the following functions allocates multiple blocks of memory, each block of the same size?  
a) malloc()  
b) realloc()  
c) calloc()  
d) free()

4. A condition where in memory is reserved dynamically but not accessible to any of the programs is called \_\_\_\_\_\_\_\_\_\_\_\_\_  
a) Memory leak  
b) Dangling pointer  
c) Frozen memory  
d) Pointer leak

5. What will happens if the statement free(a) is removed in the following C code?

#include<stdio.h>

#include<stdlib.h>

main()

{

int \*a;

a=(int\*)malloc(sizeof(int));

\*a=100;

printf("\*a%d",\*a);

free(a);

a=(int\*)malloc(sizeof(int));

\*a=200;

printf("a%p",a);

\*a=200;

printf("a%d",\*a);

}

a) Error  
b) Memory leak  
c) Dangling pointer arises  
d) 200 is printed as output

6. The incorrect statement with respect to dangling pointers is \_\_\_\_\_\_\_\_\_\_\_  
a) Pointer pointing to non-existent memory location is called dangling pointer  
b) When a dynamically allocated pointer references the original memory after it has been freed, a dangling pointer arises  
c) If memory leak occurs, it is mandatory that a dangling pointer arises  
d) Dangling pointer may result in segmentation faults and potential security risks

7. What will be the output of the following C code?

#include<stdio.h>

#include<stdlib.h>

void main()

{

char \*p = calloc(100, 1);

p = "welcome";

printf("%s**\n**", p);

}

a) error  
b) welcome  
c) memory location stored by the pointer  
d) junk value

8. In the function realloc(), if the new size of the memory block is larger than the old size, then the added memory \_\_\_\_\_\_\_\_\_\_\_  
a) is initialized to junk values  
b) is initialized to zero  
c) results in an error  
d) is not initialized

9. The free() function frees the memory state pointed to by a pointer and returns \_\_\_\_\_\_\_\_\_\_\_  
a) the same pointer  
b) the memory address  
c) no value  
d) an integer value

10. The following C code is an example of \_\_\_\_\_\_\_\_\_\_

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

main()

{

char \*p,\*q;

p=(char\*)malloc(3\*sizeof(char));

q=p;

strcpy(p,"hello");

printf("p=%s",p);

printf("q=%s",q);

free(q);

q=NULL;

gets(p);

gets(q);

printf("%s",p);

printf(“%s”,q);

}

a) Memory leak  
b) Dangling pointer  
c) Static memory allocation  
d) Linked list

**DMA Functions, Memory Leak, Dangling Pointers – 2:**

1. What will be the output of the following C code if it is executed on a 32 bit processor?

#include<stdio.h>

#include<stdlib.h>

int main()

{

int \*p;

p = (int \*)malloc(20);

printf("%d**\n**", sizeof(p));

free(p);

return 0;

}

a) 2  
b) 4  
c) 8  
d) Junk value

2. The number of arguments taken as input which allocating memory dynamically using malloc() is \_\_\_\_\_\_\_\_\_\_\_  
a) 0  
b) 1  
c) 2  
d) 3

3. Suppose we have a one dimensional array, named ‘x’, which contains 10 integers. Which of the following is the correct way to allocate memory dynamically to the array ‘x’ using malloc()?  
a) x=(int\*)malloc(10);  
b) x=(int\*)malloc(10,sizeof(int));  
c) x=malloc(int 10,sizeof(int));  
d) x=(int\*)malloc(10\*sizeof(int));

4. what will be the error (if any) in the following C code?

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

int main()

{

char \*p;

\*p = (char)calloc(10);

strcpy(p, "HELLO");

printf("%s", p);

free(p);

return 0;

}

a) No error  
b) Error in the statement: strcpy(p,”HELLO”);  
c) Error in the statement: \*p=(char)calloc(10);  
d) Error in the statement: free(p);

5. If malloc() and calloc() are not type casted, the default return type is \_\_\_\_\_\_\_\_\_\_\_  
a) void\*  
b) void\*\*  
c) int\*  
d) char\*

6. Pick out the correct statement with respect to the heap.  
a) Local variables are stored on the heap  
b) Static variables are stored on the heap  
c) Heap is the data structure that is used to implement recursive function calls  
d) Everything on the heap is anonymous

7. What will be the output of the following C code? (Given that the size of array is 4 and new size of array is 5)

#include<stdio.h>

#include<stdlib.h>

main()

{

int \*p,i,a,b;

printf("Enter size of array");

scanf("%d",&a);

p=(int\*)malloc(a\*sizeof(int));

for(i=0;i<a;i++)

printf("%d**\n**",i);

printf("Enter new size of array");

scanf("%d",&b);

realloc(p,b);

for(i=0;i<b;i++)

printf("%d**\n**",i);

free(p);

}

a)

1234

12345

b) Error  
c)

0123

01234

d)

0123

12345

8. When the pointer is NULL, then the function realloc is equivalent to the function \_\_\_\_\_\_\_\_\_\_\_  
a) malloc  
b) calloc  
c) free  
d) alloc

9. Garbage collector frees the programmer from worrying about \_\_\_\_\_\_\_\_\_\_\_  
a) Dangling pointers  
b) Creating new objects  
c) Memory leak  
d) Segmentation errors

10. If the space in memory allocated by malloc is not sufficient, then an allocation fails and returns \_\_\_\_\_\_\_\_\_\_\_  
a) NULL pointer  
b) Zero  
c) Garbage value  
d) The number of bytes available

**Multiple Choice Questions on C Preprocessor**

**Stringizers:**

1. Which of the following is a stringizing operator?  
a) < >  
b) #  
c) %  
d) ##

2. What will be the output of the following C code?

advertisement

#define sanfoundry(s,n) #s #n

main()

{

printf(sanfoundry(hello,world));

}

a) sanfoundry(hello,world)  
b) sanfoundry  
c) hello,world  
d) helloworld

3. What will be the output of the following C code?

#define display(text) printf(#text "@")

main()

{

display(hello.);

display(good morning!);

}

a) hello.@good morning!  
b) error  
c) hello.good morning!@  
d) hello.@good morning!@

4. What will be the output of the following C code?

#define display(a) #a

main()

{

printf(display("56#7"));

}

a) Error  
b) “56#7”  
c) 56#7  
d) 567

5. What will be the output of the following C code?

#define HELLO(a) #a

main()

{

printf(HELLO(good morning));

}

a) good morning  
b) goodmorning  
c) good morning  
d) error

6. What will be the output of the following C code?

#include <stdio.h>

#define sanfoundry(x) #x

int main()

{

int marks=100;

printf("value of %s is = %d**\n**",sanfoundry(marks),marks);

return 0;

}

a) error  
b) value of marks=100  
c) value of=100  
d) 100

7. What will be the output of the following C code?

#define hello(c) #c

main()

{

printf(hello(i,am));

}

a) i,am  
b) iam  
c) i am  
d) error

8. What will be the output of the following C code?

#define hello(c,d) #c #d

main()

{

printf(hello(i,"am"));

}

a) iam  
b) i“am”  
c) am  
d) “am”

9. What will be the output of the following C code?

#define F abc

#define B def

#define FB(arg) #arg

#define FB1(arg) FB(arg)

main()

{

printf(FB(F B));

FB1(F B);

}

a) F B  
b) Error  
c) FB  
d) “FB”

10. What will be the output of the following C code?

#define display(text) "$" #text

main()

{

printf(display(hello world));

}

a) hello world  
b) $helloworld  
c) $hello world  
d) error

**Conditional Preprocessor Directives – 1:**

1. What will be the output of the following C code?

#include<stdio.h>

#define max 100

main()

{

#ifdef max

printf("hello");

}

a) 100  
b) hello  
c) “hello”  
d) error

2. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ is the preprocessor directive which is used to end the scope of #ifdef.  
a) #elif  
b) #ifndef  
c) #endif  
d) #if

3. What will be the output of the following C code?

#include<stdio.h>

void main()

{

#ifndef max

printf("hello");

#endif

printf("hi");

}

a) hello  
b) hellohi  
c) error  
d) hi

4. What will be the output of the following C code?

#include<stdio.h>

#define san 557

int main()

{

#ifndef san

printf("yes");

#endif

printf("no");

}

a) error  
b) yes  
c) no  
d) yesno

5. The preprocessor directive which checks whether a constant expression results in a zero or non-zero value \_\_\_\_\_\_\_\_\_\_  
a) #if  
b) #ifdef  
c) #undef  
d) #ifndef

6. What will be the output of the following C code?

#include<stdio.h>

#define max 100

void main()

{

#if(max%10)

printf("san");

#endif

printf("foundry");

}

a) error  
b) san  
c) foundry  
d) sanfoundry

7. The preprocessor directive which is used to remove the definition of an identifier which was previously defined with #define?  
a) #ifdef  
b) #undef  
c) #ifndef  
d) #def

8. What will be the output of the following C code?

#include<stdio.h>

#define hello 10

void main()

{

printf("%d",hello);

#undef hello

printf("%d",hello);

}

a) 10  
b) hello  
c) error  
d) 1010

9. What will be the output of the following C code?

#include <stdio.h>

#define a 2

main()

{

int r;

#define a 5

r=a\*2;

printf("%d",r);

}

a) 10  
b) 4  
c) 2  
d) 5

10. What will be the output of the following C code if the value of ‘p’ is 10 and that of ‘q’ is 15?

#include<stdio.h>

int main()

{

int p,q;

printf("Enter two numbers**\n**");

scanf("%d",&p);

scanf("%d",&q);

#if(4<2)

printf("%d",p);

#elif(2>-1)

printf("%d",q);

#else

printf("bye");

#endif

}

a) 10  
b) 15  
c) bye  
d) error

**Conditional Preprocessor Directives – 2:**

1. What will be the output of the following C code?

#include<stdio.h>

#define san 10

main()

{

#ifdef san

#define san 20

#endif

printf("%d",san);

}

a) 10  
b) 20  
c) Error  
d) 1020

2. What will be the output of the following C code?

#include<stdio.h>

#define hello

main()

{

#ifdef hello

#define hi 4

#else

#define hi 5

#endif

printf("%d",hi);

}

a) 4  
b) 5  
c) 45  
d) error

3. The purpose of the preprocessor directive #error is that \_\_\_\_\_\_\_\_\_\_\_\_  
a) It rectifies any error present in the code  
b) It rectifies only the first error which occurs in the code  
c) It causes the preprocessor to report a fatal error  
d) It causes the preprocessor to ignore an error

4. What will be the output of the following C code?

#include<stdio.h>

#define max 20

main()

{

#ifndef max

#define min 10

#else

#define min 30

#endif

printf("%d",min);

}

a) 10  
b) 20  
c) 30  
d) error

5. What will be the output of the following C code?

#include<stdio.h>

#define hello 10

main()

{

#ifdef hello

#undef hello

#define hello 100

#else

#define hello 200

#endif

printf("%d",hello);

}

a) Error  
b) 10  
c) 100  
d) 200

6. What will be the output of the following C code?

#include<stdio.h>

#define sf 10

main()

{

if(sf==100)

printf("good");

else

{

printf("bad");

sf=100;

}

printf("%d",sf);

}

a) 100  
b) bad  
c) 10  
d) error

7. What will be the output of the following C code?

#include<stdio.h>

void f()

{

#define sf 100

printf("%d",sf);

}

int main()

{

#define sf 99;

f();

printf("%d",sf);

}

a) error  
b) 100  
c) 99  
d) 10099

8. What will be the output of the following C code?

#include<stdio.h>

#define INDIA 1

#define US 2

#define CC US

main()

{

#if CC==INDIA

printf("Rupee");

#elif CC==US

printf("Dollar");

#else

printf("Euro");

#endif

}

a) Euro  
b) Rupee  
c) Dollar  
d) Error

9. What will be the output of the following C code?

#define sqr(x) x\*x

main()

{

int a1;

a1=25/sqr(5);

printf("%d",a1);

}

a) 25  
b) 1  
c) 5  
d) error

10. Which of the following is not a preprocessor directive?  
a) #error  
b) #pragma  
c) #if  
d) #ifelse

**Token Concatenation:**

1. Which of the following operators is used to concatenate two strings without space?  
a) #  
b) < >  
c) \*\*  
d) ##

2. What will be the output of the following C code?

advertisement

#include <stdio.h>

#define p( n ) printf( "t" #n " = %d", t##n )

int t3=10;

int main()

{

p(3);

}

a) t=10  
b) t3=10  
c) t10=3  
d) t=3

3. What will be the output of the following C code?

#include <stdio.h>

#define p( n,m ) printf( "%d", m##n )

int main()

{

p(3,4);

}

a) Error  
b) Junk value  
c) 34  
d) 43

4. What will be the output of the following C code?

#include <stdio.h>

#define p( n,m ) printf( "%d", m##n )

#define q(a,b) printf("%d",a##b)

main()

{

p(3,4);

q(5,6);

}

a) 4356  
b) 3456  
c) 4365  
d) 3465

5. The following C code results in an error.

#include <stdio.h>

#define world( n ) printf( "t^^" #n" = %c", t##n )

int t3=1;

int main()

{

world(3);

}

a) True  
b) False

6. What will be the output of the following C code?

#include <stdio.h>

#define display( n ) printf( "a" #n " = %d", a##n )

int main()

{

display(3);

}

a) a3  
b) 31  
c) a 3  
d) error

7. What will be the output of the following C code?

#include <stdio.h>

#define hello( n ) a##n

int a3;

int main()

{

int x;

x=hello(3);

if(x!=0)

printf("hi");

else

printf("good");

}

a) error  
b) a3  
c) good  
d) hi

8. What will be the output of the following C code?

#include <stdio.h>

#define hello( n ) printf( "a" #n "= %d", a##n )

int a3=3;

int main()

{

#ifdef a3

hello(3);

#else

printf("sorry");

#endif

}

a) a3=3  
b) error  
c) a=3  
d) sorry

9. What will be the output of the following C code?

#include <stdio.h>

#define p( n ) printf( "t\*" #n " = %s", t##n )

char tsan[]="tsan";

int main()

{

int x;

x=p(san);

}

a) error  
b) tsan=tsan  
c) t\*san=t\*san  
d) t\*san=tsan

10. What will be the output of the following C code?

#include <stdio.h>

#define p( n ) printf( "t%%\n" #n " = %d", t##n )

int t3=10;

int main()

{

int x;

x=p(3);

}

a)

t%

3=10

b) t3=10  
c) t%3=10  
d)

t

%3=10

**Miscellaneous Topics in C**

**Inline:**

1. Name the function whose definition can be substituted at a place where its function call is made \_\_\_\_\_\_\_\_\_  
a) friends function  
b) inline function  
c) volatile function  
d) external function

2. What will be the output of the following C code?

advertisement

#include <stdio.h>

void **inline** func1(int a, int b)

{

printf ("a=%d and b=%d**\n**", a, b);

}

int **inline** func2(int x)

{

return x\*x;

}

int main()

{

int tmp;

func1(1,4);

tmp = func2(6);

printf("square val=%d**\n**", tmp);

return 0;

}

a)

a=1 and b=4

square val = 36

b) a=4 and b=1  
c) error  
d) square val = 36

3. What will be the error (if any) in the following C code?

#include <stdio.h>

void **inline** func1(float b)

{

printf ("%lf**\n**",b\*2);

}

int main()

{

**inline** func1(2.2);

return 0;

}

a) No error  
b) Error in statement: void inline func1(float b)  
c) Error in statement: printf(“%lf\n”,b\*2);  
d) Error in statement: inline func1(2.2);

4. What will be the output of the following C code?

#include <stdio.h>

void **inline** f1(char b)

{

printf ("%d**\n**",b);

}

int main()

{

f1('a');

return 0;

}

a) a  
b) 65  
c) error  
d) 97

5. What will be the output of the following C code?

#include <stdio.h>

void **inline** func1(char b[10])

{

printf ("%c**\n**",b[2]);

}

int main()

{

func1("sanfoundry");

return 0;

}

a) s  
b) n  
c) a  
d) error

6. The following C code results in an error. State whether this statement is true or false.

#include <stdio.h>

void f(double b)

{

printf ("%ld**\n**",b);

}

int main()

{

**inline** f(100.56);

return 0;

}

a) True  
b) False

7. What will be the output of the following C code?

#include<stdio.h>

static **inline** int max(int a, int b)

{

return a > b ? a : b;

}

main()

{

int m;

m=max(-6,-5);

printf("%d",m);

}

a) -6  
b) -5  
c) Junk value  
d) Error

8. What will be the output of the following C code?

#include<stdio.h>

#define inline

**inline** f(char a)

{

#ifdef inline

printf("%c",a);

#endif

}

main()

{

f('a');

}

a) Error  
b) a  
c) No error but nothing will be printed as output  
d) 97

9. What will be the output of the following C code?

#include<stdio.h>

**extern** **inline** int min(int a, int b)

{

return a < b ? a : b;

}

main()

{

int m;

m=min(3,-5);

printf("%d",m);

}

a) Error  
b) 3  
c) -5  
d) Junk value

10. To have GCC inline the given function regardless of the level of optimization, we must declare the function with the attribute \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
a) optimize\_inline  
b) packed\_inline  
c) always\_inline  
d) level\_inline

**Endianness:**

1. A machine in which the least significant byte is stored in the smallest address is \_\_\_\_\_\_\_\_\_\_  
a) Big endian machine  
b) Bi-endian machine  
c) Binary bit machine  
d) Little endian machine

2. If the output of the following C code is “Big endian”, then what will be the value of \*a is?

advertisement

#include <stdio.h>

int main()

{

unsigned int i = 1;

char \*a = (char\*)&i;

if (\*a)

printf("Little endian");

else

printf("Big endian");

getchar();

return 0;

}

a) -1  
b) 0  
c) 1  
d) 2

3. It is possible for a processor to support both little and big endian methods.  
a) True  
b) False

4. The standard byte order for networks is \_\_\_\_\_\_\_\_\_\_\_\_  
a) Bit-Binary  
b) Little endian  
c) Big endian  
d) Bi-endian

5. Which of the following is not an example of big endian machines?  
a) Power PC  
b) Motorola 68K  
c) SPARC processors  
d) ARM processors

6. Suppose we transfer a file written on a little endian machine to a big endian machine and there is no transformation from little endian to big endian, then what happens?  
a) The big endian machine throws an error when it receives the file  
b) The big endian machine reads the file in the reverse order  
c) The big endian machine does not read the file  
d) The big endian machine reads the file in the normal order  
View Answer

7. File formats which have \_\_\_\_\_\_\_\_\_ as a basic unit are independent of endianness.  
a) 1 byte  
b) 2 bytes  
c) 3 bytes  
d) 4 bytes

8. If the code shown below is executed on a little endian machine, then what will be the output of the following C code?

#include<stdio.h>

main()

{

int y=1;

printf("%d", (\*(char\*)&y));

}

a) 1  
b) 1000  
c) 9999  
d) 0

9. If the data “ABCD” is to be stored in a little endian machine, it will be stored as \_\_\_\_\_\_\_\_\_  
a) ABCD  
b) DCBA  
c) CDAB  
d) BCDA

10. The sequential order used to interpret a range of bytes in the memory of a computer is known as \_\_\_\_\_\_\_\_\_  
a) Ordering  
b) Sequencing  
c) Endianness  
d) Byte prediction

**Recursion:**

1. What will be the output of the following C code?

#include<stdio.h>

main()

{

int n;

n=f1(4);

printf("%d",n);

}

f1(int x)

{

int b;

if(x==1)

return 1;

else

b=x\*f1(x-1);

return b;

}

a) 24  
b) 4  
c) 12  
d) 10

2. The data structure used to implement recursive function calls \_\_\_\_\_\_\_\_\_\_\_\_\_  
a) Array  
b) Linked list  
c) Binary tree  
d) Stack

3. The principle of stack is \_\_\_\_\_\_\_\_\_\_  
a) First in first out  
b) First in last out  
c) Last in first out  
d) Last in last out

4. In the absence of a exit condition in a recursive function, the following error is given \_\_\_\_\_\_\_\_\_\_  
a) Compile time error  
b) Run time error  
c) Logical error  
d) No error

5. What will be the output of the following C code?

#include<stdio.h>

main()

{

int n,i;

n=f(6);

printf("%d",n);

}

f(int x)

{

if(x==2)

return 2;

else

{

printf("+");

f(x-1);

}

}

a) ++++2  
b) +++++2  
c) +++++  
d) 2

6. How many times is ‘a’ printed when the following C code is executed?

#include<stdio.h>

main()

{

int a;

a=f1(10);

printf("%d",a);

}

f1(int b)

{

if(b==0)

return 0;

else

{

printf("a");

f1(b--);

}

}

a) 9 times  
b) 10 times  
c) 0 times  
d) Infinite number of times

7. What will be the output of the following C code?

#include<stdio.h>

int main()

{

int n=10;

int f(int n);

printf("%d",f(n));

}

int f(int n)

{

if(n>0)

return(n+f(n-2));

}

a) 10  
b) 80  
c) 30  
d) Error

8. What will be the output of the following C code?

#include<stdio.h>

int main()

{

printf("Hello");

main();

return 0;

}

a) Hello is printed once  
b) Hello infinite number of times  
c) Hello is not printed at all  
d) 0 is returned

9. What will be the output of the following C code if the input given to the code shown below is “sanfoundry”?

#include<stdio.h>

#include<stdio.h>

#define NL '\n'

void main()

{

void f(void);

printf("enter the word**\n**");

f();

}

void f(void)

{

char c;

if((c=getchar())!=NL)

{

f();

printf("%c",c);

}

return;

}

a) sanfoundry  
b) infinite loop  
c) yrdnuofnas  
d) fnasyrdnuo

10. Iteration requires more system memory than recursion.  
a) True  
b) False

**Signed Qualifier:**

1. In a signed integer, the sign is represented by \_\_\_\_\_\_\_\_\_\_\_  
a) Least significant bit  
b) Most significant bit  
c) System dependent  
d) The mean of the most significant bit and the least significant bit

2. Sign qualifiers can be applied to double.  
a) True  
b) False

3. What will be the output of the following C code?

#include<stdio.h>

int main()

{

signed char ch= ‘a’;

printf(“%u”,ch);

return 0;

}

a) -65  
b) 65  
c) -97  
d) 97

4. What will be the output of the following C code?

#include<stdio.h>

main()

{

signed char a[]= “BAT”;

printf(“%c”, a[1]);

return 0;

}

a) -A  
b) BAT  
c) A  
d) 65

5. What is the default state of an integer in c language?  
a) Signed  
b) Unsigned  
c) System dependent  
d) There is no default state

6. Suppose we have: int a=100; (signed by default).  
If we want to convert this to an unsigned long integer, we can do this by making the following small change:  
a) int a=lu100;  
b) int a= 100ul;  
c) int a=uu100;  
d) int a=100uu;

7. What is the binary representation of the integer -14?  
a) 11110  
b) 01110  
c) 01100  
d) 11100

8. Which of the following header files must necessarily be included in your code, if you want to find the minimum value of unsigned short integer?  
a) stdio.h  
b) stdlib.h  
c) limits.h  
d) math.h

9. What will be the error in the following C code?

main()

{

long float a=-25.373e22;

printf("%lf",a);

}

a) Negative number cannot be assigned to float data type  
b) Long and float cannot be used together  
c) Does not result in error  
d) Logical error

10. What will be the output of the following C code?

main()

{

unsigned a=10;

long unsigned b=5l;

printf(“%lu%u”,a,b);

}

a) 105  
b) 510  
c) 10  
d) error