SECTION A Multiple Choice

Instruction: Write the letter of the correct answer in the box provided. Each correct answer is worth 1 mark.

1. Which phase in the compilation process translates a compilation unit into an assembler file?

   (a) Preprocessing

   (b) Compilation

   (c) Assembly

   (d) Linking

   $\boxed{\text{b}}$

2. Which of the following is an **invalid** integer literal?

   (a) `1234`

   (b) `0xbeer`

   (c) `-100U`

   (d) `0234`

   $\boxed{\text{b}}$

3. Which of the following is a **valid** C identifier?

   (a) `1node`

   (b) `break`

   (c) `first-last-name`

   (d) None of the above

   $\boxed{\text{d}}$

4. A C program contains the following declarations:

```
int i, j;
long ix;
short s;
float x;
```

   What is the resulting data type of the following expression?

```
(int) ix / s - 2.6 * c + x * i / j ?
```

(a) `int`
(b) `long`
(c) `float`
(d) `double`

d

5. What value is assigned to `j` in the expression `j = ++i % i - 1` when `i = 3`?
   (a) 2
   (b) 1
   (c) 0
   (d) -1

d

6. Consider the following C code snippet:

```
int i = 1, j = 2, k = 10;
i = j += k / 2;
```

What is the value of `i` after the second statement?
(a) 2
(b) 5
(c) 7
(d) 8

c

7. In call by value, the values of formal parameters are copied to actual parameters.
   (a) True
   (b) False

b

8. Consider the following function-like macro:

```
#define MACRO(X, Y)  X * Y - X / Y
```

What value does the macro evaluate to when invoked as `MACRO(2+6, 4-2)`?

(a) 12

(b) 15

(c) 18

(d) 21

d

9. Consider the following statement:

```
int array[10] = {1, 2, 3, 4, 5};
```

Which of the following statements are **valid**?

i. `array` has 5 elements.
ii. The number of elements in `array` can be obtained by `sizeof(array)`.
iii. The value of `array[0]` is 1.
iv. The value of `array[5]` is 5.

(a) i and ii

(b) iii

(c) iv

(d) They are all invalid

b

10. Consider the following statement:

```
char str[12] = "Twelve";
```

What is the length of the string `str`?

(a) 6

(b) 7

(c) 11

(d) 12

a

11. Given the declaration below:

```
char name[30];
```

Which of the following statements are **valid**?

i. `strcpy(name, "Alice");`
ii. `name = "Bobby";`
iii. `name = {'C', 'a', 'k', 'e'};`
iv. `name[0] = 'D';`

(a) i and iii
(b) i and iv
(c) ii and iii
(d) They are all valid

b

12. Consider the following C code snippet:

```
char str1[] = "String 1";
char *str2 = "String 2";
```

Which of the following statements involving `str1` and `str2` are **valid**?

i. `str1[0] = 's';`
ii. `str2[0] = 's';`
iii. `strcpy(str1, str2);`
iv. `strcpy(str2, str1);`
v. `str2 = str1;`
vi. `str1 = str2;`

(a) i, iii, and v
(b) ii, iv, and vi
(c) i and ii
(d) They are all valid

a

13. Consider the following declarations:

```
int n[] = {1, 2, 3, 4};
int *p = n + 1;
```

What is the value of `p[1]`?

(a) 1

(b) 2

(c) 3

(d) 4

c

14. Using the same declarations from question (13), what is the value of `*p+1`?

(a) 1

(b) 2

(c) 3

(d) 4

c

15. Consider the following C code snippet:

```
enum { apple, banana, cherry=5, date } myfruit = banana;
```

What is the value of `myfruit`?

(a) 0

(b) 1

(c) 2

(d) 6

b

16. Which of the following statements regarding storage classes are **valid**?

i. A variable with static storage class will always have global scope.

ii. A variable that is declared outside any function will be stored in the data segment.

iii. The initial value of an uninitialized automatic variable is garbage.

iv. A variable with register storage class will always be stored in a register.
v. Variables stored in the heap segment will have static lifetime.

(a) i and ii
(b) ii and iii
(c) iv and v
(d) The are all valid

b

17. Which of the following is equivalent to the call `malloc(10 * sizeof(double))`?
    (a) `calloc(10*sizeof(double))`
    (b) `calloc(10)`
    (c) `calloc(sizeof(double))`
    (d) `calloc(10, sizeof(double))`

d

18. Which of the following statements regarding memory leak are **valid**?

    i. Program will not be able to access leaked memory.
    ii. Leaked memory will no longer be in the heap segment.
    iii. Leaked memory cannot be freed, potentially causing program memory usage to keep on growing.
    iv. Leaked memory is automatically freed using garbage collection.
    v. Every instance of memory leak will always result in undefined program behaviour.

    (a) i and iii
    (b) ii and iv
    (c) i, iii and v
    (d) ii, iv and v

a

19. Consider the following code snippet:

```
char *ptr = (char *)malloc(16*sizeof(char));
realloc(ptr, 32*sizeof(char));
```

After the call to `realloc()` on the second line, `ptr` still points to the previously allocated memory on the the first line.

(a) True
(b) False

a

20. Consider the following code snippet:

```
union {
    char c;
    short s;
    long l;
} u;

u.c = 'A';
```

What is the size of the variable `u` be equal to?

(a) `sizeof(char)`

(b) `sizeof(short)`

(c) `sizeof(long)`

(d) None of the above

c

SECTION B Short Answer

Instruction: Write your answer in the box provided.

21. Declare a macro symbolic constant `CHARGE` with a single-precision floating point value 1.602×10^{-19}. **(2 marks)**

```
#define CHARGE 1.602e-19f
```

22. Consider the following C program: **(2 marks)**

```
#include <stdio.h>

int func(int a, int b)
{
    return --a * b;
}

int main(void)
{
    int i = 5;
    int j = 2 * func(1+2, i+1);
    printf("%d %d", i, j);
    return 0;
}
```

What is the output of the program?

```
5 24
```

23. Re-write `func(int a, int b)` in program in question 22 into a function-like macro `FUNC(A, B)`, such that when the call to `func(1+2, i+1)` in the program is replaced with `FUNC(1+2 ,i+1)`, the outputs will remain the same. **(2 marks)**

```
#define FUNC(A,B) (((A)-1)*(B))
```

24. Using only one C statement, declare an array which can hold 1000 integers named `intarray` with initial values 1, 2, 3 and 4 for the first four elements, and 0 for the remaining elements: **(2 marks)**

```
int intarray[1000] = {1, 2, 3, 4};
```

25. Given the following array and pointer declarations: **(3 marks)**

```
int iarray[] = {1,2,3,4,5};
int *ip = array;
```

Write 3 C expressions showing 3 different ways to access the value stored in the **first element** of `iarray`.

Any 3 of the following: `iarray[0]`, `*iarray`, `*ip`, `ip[0]`

26. Declare an enumeration type with identifiers `low`, `medium`, and `high` having values of 10, 11, and 12, respectively. Use `risk_level` as tag of the enumeration type. **(3 marks)**

```
enum risk_level { low = 10, medium, high };
```

27. Given the following variable declarations:

```
int a[] = {1, 2, 3, 4, 5};
int *ip = a;
```

Suppose that an `int` occupies 4 bytes in memory. The array `a` is at memory address 600, while `ip` is at memory address 500 (all addresses are in decimal).

(a) What is the numeric value of the expression `a`? **(1 mark)**

600

(b) What is the numeric value of the expression `ip+1`? **(1 mark)**

600+1*4 = 604

(c) What is the numeric value of the expression `&a[2]`? **(1 mark)**

600+2*4 = 608

(d) What is the numeric value of the expression `*(ip+1)`? **(1 mark)**

2

(e) What is the numeric value of the expression `*++ip`? **(1 mark)**

2

28. Consider the following C code snippet: **(1 mark)**

```
char *cp;
cp = (char *)malloc(15*sizeof(char));
```

Assuming that the allocation is successful, what is the size (in bytes) of the memory block pointed to by `cp`?

15

29. Briefly explain why a function that returns an address to an automatic variable is a problem. **(2 marks)**

An automatic variable only exists within the function. Therefore, returning an address to this variable would cause undefined program behaviour.

30. Consider a singly-linked list which contains a list of integers. A node in this list is defined as follows:

```
typedef struct node {
    int data;
    struct node *next;
} Node;
```

Suppose that the node `head` points to the head of the list. Suppose further the list contains the integers 4, 2, 7, 9, and 6, where 4 is at the head of the list.

(a) What is the value of `head->data`? **(1 mark)**

4

(b) What is the value of `head->next->next->data`? **(1 mark)**

7

(c) What is the output of the following code snippet? **(1 mark)**

```
Node *p = head->next;
while(p != NULL) {
    printf("%d", p->data);
    p = p->next;
}
```

2796

* * * * * * * * * * * * * * *