1. What is the primary purpose of a hash function?

A) To sort elements in an array.

**B) To convert data into a fixed-size value.**

C) To allocate memory dynamically.

D) To encrypt data for secure storage.

2. #include <iostream>

#include <unordered\_map>

using namespace std;

int main() {

unordered\_map<string, int> hashTable;

hashTable["apple"] = 5;

hashTable["banana"] = 3;

hashTable["cherry"] = 8;

cout << hashTable["banana"] << endl;

return 0;

}

What will be the output of the program?

A) 5

**B) 3**

C) 8

D) It will result in a compilation error.

3.What is a collision in the context of hash tables?

A) The process of resizing a hash table.

**B) A situation where two different keys map to the same hash value.**

C) The act of inserting an element into a hash table.

D) The process of deleting an element from a hash table.

4. #include <iostream>

#include <unordered\_map>

using namespace std;

int main() {

unordered\_map<string, int> hashTable;

hashTable["apple"] = 5;

hashTable["banana"] = 3;

hashTable["cherry"] = 8;

cout << hashTable["orange"] << endl;

return 0;

}

What will be the output of the program?

**A) 0**

B) 3

C) 8

D) It will output a default value (0) and insert "orange" into the hash table.

5. Which property of a good hash function helps distribute keys evenly across the hash table?

A) Collisions

B) Clustering

**C) Uniformity**

D) Indexing

6. #include <iostream>

#include <unordered\_map>

using namespace std;

int main() {

unordered\_map<string, int> hashTable;

hashTable["apple"] = 5;

hashTable["banana"] = 3;

hashTable["cherry"] = 8;

hashTable.erase("banana");

cout << hashTable["banana"] << endl;

return 0;

}

What will be the output of the program?

A) 5

**B) 0**

C) 8

D) It will result in a compilation error.

7. What is the load factor of a hash table?

A) The number of elements stored in the table.

B) The size of the table.

**C) The ratio of occupied slots to total slots in the table.**

D) The number of collisions in the table.

8. Which of the following hash functions is most likely to cause clustering in a hash table?

**A) h(k) = k % m**

B) h(k) = floor(m \* (kA mod 1))

C) h(k) = k

D) h(k) = ((k / m) + k \* m) + k % m

9. Which collision resolution technique involves creating linked lists at each slot?

A) Linear Probing

B) Quadratic Probing

**C) Separate Chaining**

D) Double Hashing

10.

#include <iostream>

#include <unordered\_map>

using namespace std;

int main() {

unordered\_map<int, string> hashTable;

hashTable[5] = "three";

hashTable[3] = "five";

hashTable[8] = "eight";

cout << hashTable[5] << endl;

return 0;

}

What will be the output of the program?

A) "five"

**B) "three"**

C) "eight"

D) It will result in a compilation error.

11. In a hash table, what is the time complexity of inserting an element with separate chaining?

**A) O(1)**

B) O(log n)

C) O(n)

D) It depends on the load factor.

12. What will be the output if we pass key as 5 to the below function?

int total\_elements=13;

int getHash(int key){

return key % total\_elements;

}

A) 2

**B) 5**

C) 3

D) 1

13. Which hashing technique requires a secondary hash function to resolve collisions?

A) Linear Probing

B) Quadratic Probing

C) Separate Chaining

**D) Double Hashing**

14. What is the operation of following function?

char\* func(HashTable\* table, char\* key)

{

int index = hash\_function(key);

Ht\_item\* item = table->items[index];

// Provide only non-NULL values.

if (item != NULL)

{

if (strcmp(item->key, key) == 0)

return item->value;

}

return NULL;

}

A) Generate hash key

**B) Search the key in hash table**

C) Insert key in hash table

D) Delete element from hash table

15. What is the worst-case time complexity of searching for an element in a hash table?

A) O(1)

B) O(log n)

**C) O(n)**

D) It depends on the load factor.

16. What is the operation of following function?

void func(int k, int v) {

int h = HashFunc(k);

while (t[h] != NULL && t[h]->k != k) {

h = HashFunc(h + 1);

}

if (t[h] != NULL)

delete t[h];

t[h] = new HashTableEntry(k, v);

}

A) Generate hash key

B) Search the key in hash table

**C) Insert key in hash table**

D) Delete element from hash table

17. What is the primary disadvantage of using open addressing for collision resolution?

A) It requires extra memory for linked lists.

**B) It leads to clustering.**

C) It cannot handle large datasets.

D) It may require frequent resizing.

18. #include <iostream>

using namespace std;

int midHash(int key, int size) {

return (key + size / 2) % size;

}

int main() {

int key = 8;

int size = 10;

int hashValue = midHash(key, size);

cout << hashValue << endl;

return 0;

}

What will be the output of the program?

A) 0

B) 4

**C) 3**

D) 8

19. Which hashing technique aims to find the next available slot when a collision occurs?

**A) Linear Probing**

B) Quadratic Probing

C) Separate Chaining

D) Double Hashing

20. #include <iostream>

using namespace std;

int midHash(int key, int size) {

return (key + size / 2) % size;

}

int main() {

int key = 12;

int size = 6;

int hashValue = midHash(key, size);

cout << hashValue << endl;

return 0;

}

What will be the output of the program?

A) 0

B) 1

**C) 3**

D) 5