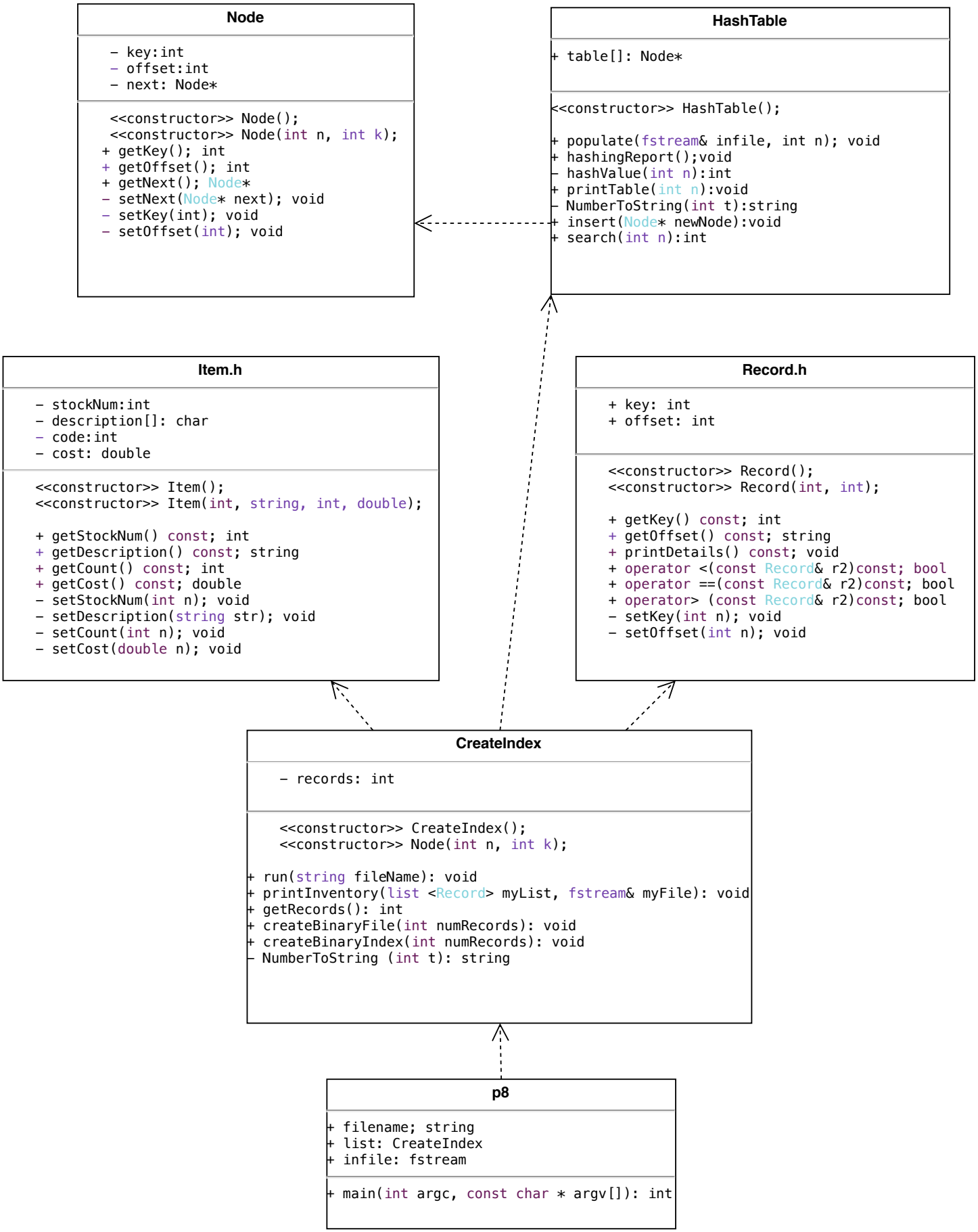
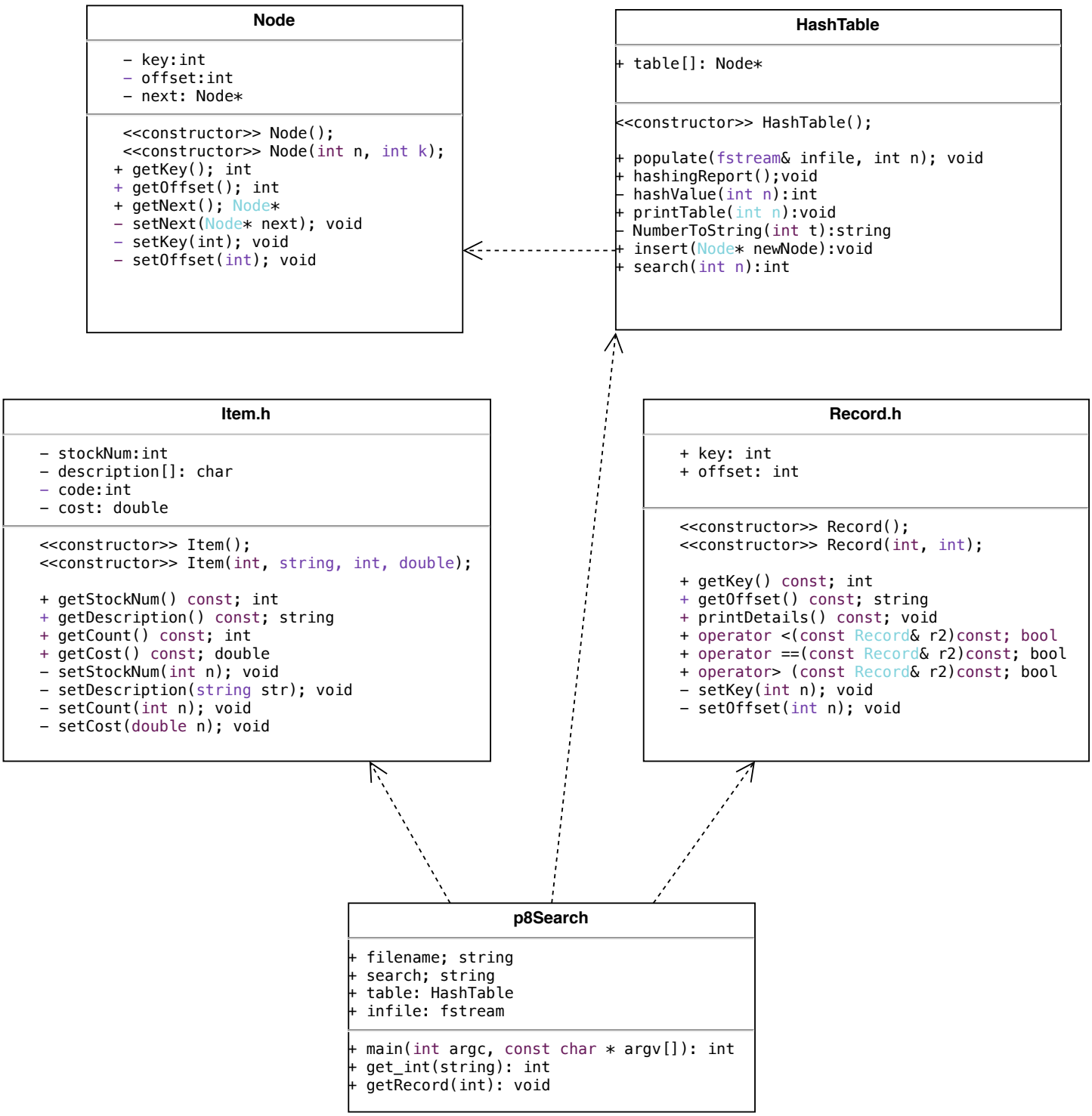

Node
- key:int
- offset:int
- next: Node*
<<constructor>> Node();
<<constructor>> Node(int n, int k);
+ getKey(); int
+ getOffset(); int
+ getNext(); Node*
- setNext(Node* next); void
- setKey(int); void
- setOffset(int); void
HashTable
+ table[]: Node*
<<constructor>> HashTable();
+ populate(fstream& infile, int n); void
+ hashingReport();void
- hashValue(int n):int
+ printTable(int n):void
- NumberToString(int t):string
+ insert(Node* newNode):void
+ search(int n):int
Item.h
- stockNum:int
- description[]: char
- code:int
- cost: double
<<constructor>> Item();
<<constructor>> Item(int, string, int, double);
+ getStockNum() const; int
+ getDescription() const; string
+ getCount() const; int
+ getCost() const; double
- setStockNum(int n); void
- setDescription(string str); void
- setCount(int n); void
- setCost(double n); void
Record.h
+ key: int
+ offset: int
<<constructor>> Record();
<<constructor>> Record(int, int);
+ getKey() const; int
+ getOffset() const; string
+ printDetails() const; void
+ operator <(const Record& r2)const; bool
+ operator ==(const Record& r2)const; bool
+ operator> (const Record& r2)const; bool
- setKey(int n); void
- setOffset(int n); void
CreateIndex
- records: int
<<constructor>> CreateIndex();
<<constructor>> Node(int n, int k);
+ run(string fileName): void
+ printInventory(list <Record> myList, fstream& myFile): void
+ getRecords(): int
+ createBinaryFile(int numRecords): void
+ createBinaryIndex(int numRecords): void
- NumberToString (int t): string
p8
+ filename; string
+ list: CreateIndex
+ infile: fstream
+ main(int argc, const char * argv[]): int


Node
- key:int
- offset:int
- next: Node*
<<constructor>> Node();
<<constructor>> Node(int n, int k);
+ getKey(); int
+ getOffset(); int
+ getNext(); Node*
- setNext(Node* next); void
- setKey(int); void
- setOffset(int); void
HashTable
+ table[]: Node*
<<constructor>> HashTable();
+ populate(fstream& infile, int n); void
+ hashingReport();void
- hashValue(int n):int
+ printTable(int n):void
- NumberToString(int t):string
+ insert(Node* newNode):void
+ search(int n):int
Item.h
- stockNum:int
- description[]: char
- code:int
- cost: double
<<constructor>> Item();
<<constructor>> Item(int, string, int, double);
+ getStockNum() const; int
+ getDescription() const; string
+ getCount() const; int
+ getCost() const; double
- setStockNum(int n); void
- setDescription(string str); void
- setCount(int n); void
- setCost(double n); void
Record.h
+ key: int
+ offset: int
<<constructor>> Record();
<<constructor>> Record(int, int);
+ getKey() const; int
+ getOffset() const; string
+ printDetails() const; void
+ operator <(const Record& r2)const; bool
+ operator ==(const Record& r2)const; bool
+ operator> (const Record& r2)const; bool
- setKey(int n); void
- setOffset(int n); void
p8Search
+ filename; string
+ search; string
+ table: HashTable
+ infile: fstream
+ main(int argc, const char * argv[]): int
+ get_int(string): int
+ getRecord(int): void